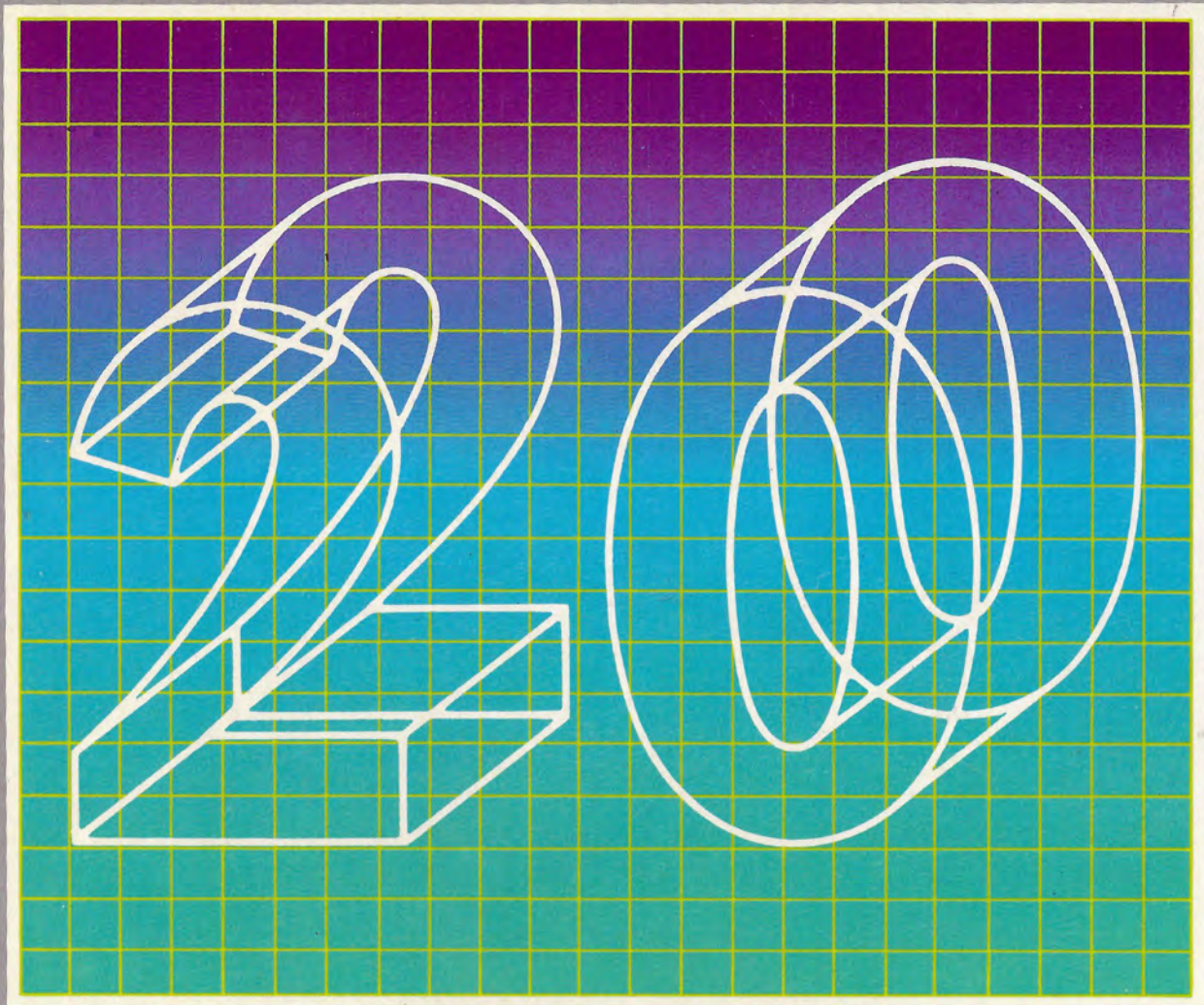


BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



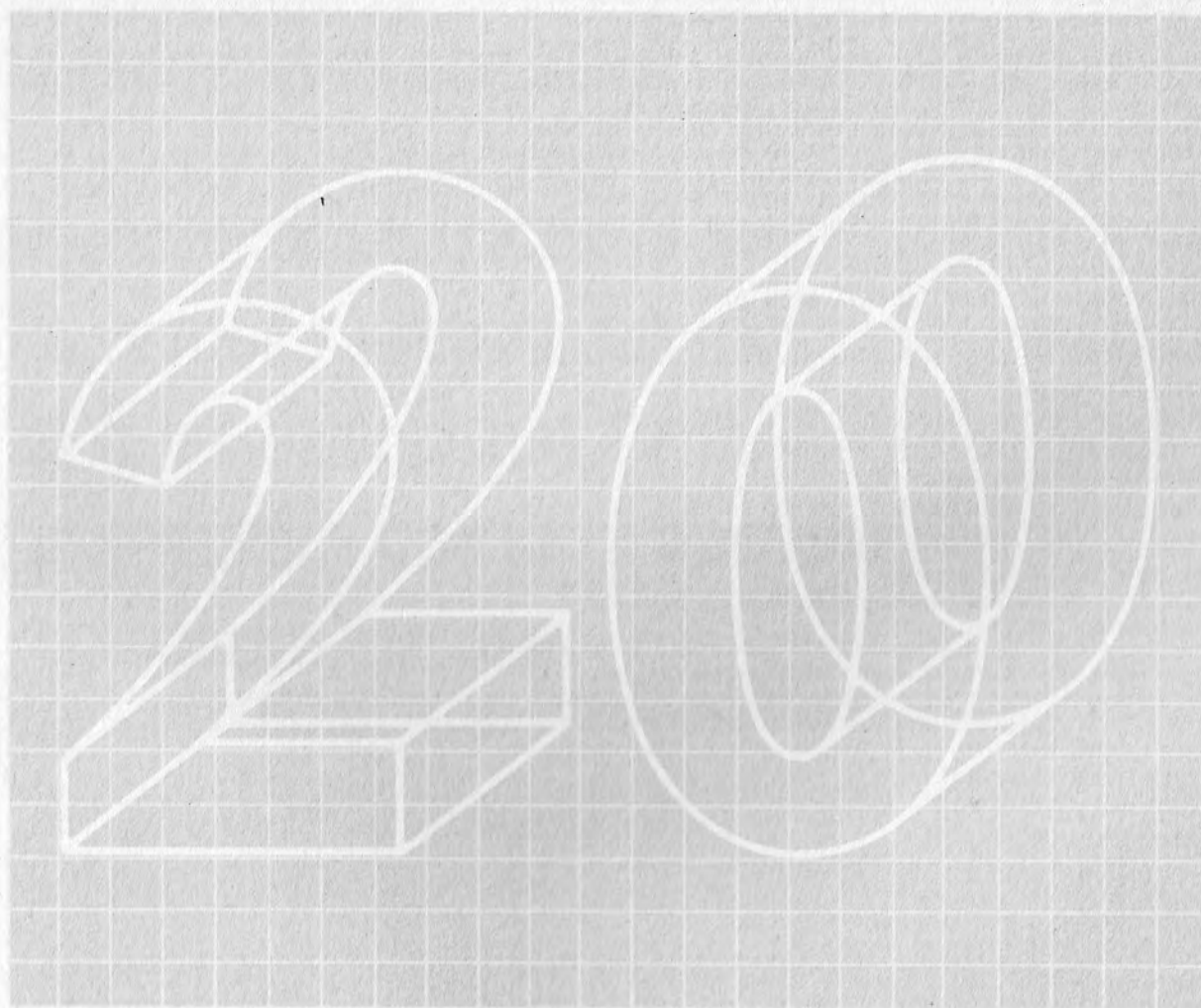
DISFRUTA TU ORDENADOR,
APRENDE Y HAZ COSAS CON EL

EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:

RICARDO ESPAÑOL CRESPO.

Gerente:

ANTONIO G. CUERPO.

Directora de producción:

MARIA LUISA SUAREZ PEREZ.

Director de la colección:

JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:

BRAVO-LOFISH.

Maquetación:

D. S. M.

Dibujos:

JOSE OCHOA

Tomo 20. «Experiencias y prácticas en logo», Carlos Albert, Técnico de Informática; Adoración Llena Jubero, Técnico de Informática. «Manejo de sprites y elementos gráficos», «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprende con el ordenador», AULA DE INFORMATICA APLICADA: José María Esteban, Diplomado en Telecomunicación; Alejandro Marcos, Licenciado en Química. «El Taller del Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», «Vocabulario de Informática», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:

Pedro Teixeira, 8-2.º. Teléf. 810 52 13. 28020 Madrid.

Publicidad:

Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:

COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.

Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:

CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.

Buenos Aires - 1.290. Argentina.

Todos los derechos reservados. Este libro no puede ser, en parte o totalmente, reproducido, memorizado en sistemas de archivo, o transmitido en cualquier forma o medio, electrónico, mecánico, fotocopia o cualquier otro, sin la previa autorización del editor.

ISBN del tomo: 84-7688-112-6

ISBN de la obra: 84-7688-047-2

Fotocomposición:

ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:

MATEU CROMO. PINTO (Madrid).

© Ediciones Siglo Cultural, S. A., 1986

Depósito legal: M-43.185-1986

Printed in Spain - Impreso en España.

Suscripciones y números atrasados:

Ediciones Siglo Cultural, S. A.

Pedro Teixeira, 8-2.º. Teléf. 810 52 13. 28020 Madrid.

Junio, 1987

P.V.P. Canarias: 355.—Ptas.

INDICE

■ EXPERIENCIA Y PRACTICAS EN LOGO	5
■ MANEJO DE SPRITES Y ELEMENTOS GRAFICOS	17
■ TRUCOS Y RUTINAS BASICAS	23
■ EL TALLER DEL HARDWARE	31
■ APRENDER CON EL ORDENADOR	41
■ PEQUEÑA HISTORIA DE LA INFORMATICA	47
■ TEMAS MONOGRAFICOS DE VANGUARDIA	51
■ TERMINOLOGIA DE VIDEOTEXTOS	55
■ VOCABULARIO DE INFORMATICA	57

EXPERIENCIA Y PRACTICAS EN LOGO

Algo más sobre los condicionales



A conocemos la forma de establecer una determinada condición dentro de un proceso realizado en Logo.

Conozcamos otras posibilidades que nos ofrecen algunas versiones de Logo, para determinar condiciones.

nes.

Una condición hay que iniciarla utilizando la orden SI seguida de una expresión, bien sea ésta una expresión de relación, una expresión lógica o una combinación de ambas.

La expresión que interviene en la condición devuelve dos posibles resultados, CIERTO o FALSO, y dependiendo del que se trate, se ejecuta una acción u otra.

La forma de expresarlo es:

SI condición [Acción 1]

Se cumple la
condición
[Acción 2]

No se cumple la
condición

Esta expresión condicional podemos desdoblarla de la siguiente forma:

SI condición CIERTA [Acción 1]

SI condición FALSA [Acción 2]

Algunas versiones de Logo permiten la posibilidad de indicar una determinada acción a tomar, preguntando directamente por el resultado que previamente ha dado una condición. Es decir, indicar que si el resultado que ha dado una condición es CIERTA, realizar una acción y si el resultado ha sido FALSO, realizar otra.

Las órdenes que nos permiten esto son:

SIC

y

SIF

abreviaturas respectivamente de:

SI Cierto

SI Falso

La forma de expresarlas es:

SIC [Acción 1]

SIF [Acción 2]

Observa que en este caso no se establece la condición, sino que se investiga sobre el resultado que ha devuelto la condición. Pero, y la condición, ¿dónde está expresada?

Para solucionar esto, conozcamos otra nueva orden Logo.

Si utilizamos las órdenes SIC y SIF, tenemos que saber previamente el resultado que ha devuelto la condición con la que estamos trabajando. Para ello utilizamos la orden:

COMPRUEBA condición

Esta orden determina si la condición establecida es cierta o falsa para utilizarla posteriormente con SIC o SIF.

Así pues, la orden SIC ejecutará las órdenes que se especifican en la primera acción, si el resultado del último COMPRUEBA es cierto, no haciendo nada en el caso contrario, y SIF ejecutará las órdenes que se especifican en la segunda acción, si el resultado del último COMPRUEBA ha sido falso, no realizando acción alguna en caso contrario.

La orden COMPRUEBA, sólo tendrá ac-

La recursividad consiste en que un procedimiento se ejecute a sí mismo.

EXPERIENCIA Y PRACTICAS EN LOGO

ción dentro del procedimiento donde se encuentra.

La forma general de expresar esta nueva forma de condicionales es:

- ? COMPRUEBA condición
- ? SIC [Acción 1]
- ? SIF [Acción 2]

El siguiente ejemplo nos demuestra la forma de utilizar correctamente todo lo que hemos visto.

— Supongamos que dependiendo del valor que tome una variable aleatoriamente entre 0 y 1, la Tortuga tiene que mostrarse u ocultarse. Si el valor de la variable es cero, la Tortuga se mostrará, y si es uno, se ocultará.

- ? PARA EJEMPLO
- > HAZ "V AZAR 2
- > COMPRUEBA :V = 0
- > SIC [MT]
- > SIF [OT]
- > FIN

Asignamos a la variable :V un valor aleatorio entre 0 y 1. Comprueba si el valor de V es igual a cero. Si lo es, obtenemos como resultado CIERTO, y si no lo es, FALSO. En el caso de que no sea cero, a la fuerza tendrá que ser uno.

A continuación, ordenamos que si el resultado de la condición es CIERTO, la Tortuga se muestre (MT), y si el resultado es FALSO, la Tortuga se OCULTE (OT).

También podíamos haber definido el procedimiento de la siguiente forma:

- ? PARA EJEMPLO
- > HAZ "V AZAR 2
- > SI :V = 0 [MT] [OT]
- > FIN

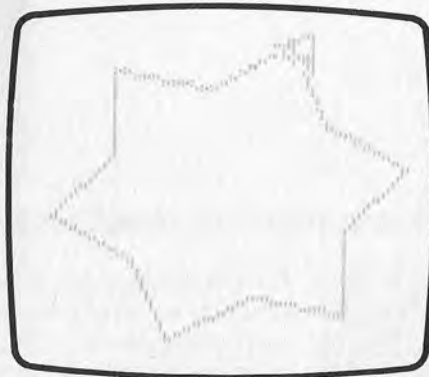
Las órdenes COMPRUEBA, SIC y SIF no existen en todas las versiones de Logo. Sí disponen de ellas las versiones de los MSX y de los PC-COMPATIBLES.

■ La recursividad

Una de las grandes posibilidades del Logo es la recursividad en los procedimientos

y consiste en que un procedimiento se llame a sí mismo.

Introduce el procedimiento ESTRELLA en tu ordenador y asómbrate.



- ? PARA ESTRELLA
- > MT
- > REPITE 6 [AV 20 GD 100 AV 20 GI 40]
- > ESTRELLA
- > FIN

Para ejecutarlo teclea:

- ? ESTRELLA

¡CUIDADO!, antes debes saber que este procedimiento mantendrá a la Tortuga dibujando estrellas eternamente, excepto que lo detengas manualmente interrumpiendo la ejecución.

Según el ordenador con el que estés trabajando, la tecla o teclas a pulsar para detener la ejecución son:

En el SPECTRUM

CAPS y BREAK/SPACE simultáneamente.

En los MSX

CTRL y STOP simultáneamente.

En los PC-compatibles

F10 o CTRL y G simultáneamente.

Vemos que dentro del procedimiento que acabamos de definir una de las órdenes es una llamada a un procedimiento, y ese procedimiento al que se llama no es otro que el que se está ejecutando.

En un procedimiento recursivo una de las órdenes es el nombre del mismo procedimiento.

Cuando la ejecución del procedimiento llegue a esa orden de llamada, el procedimiento que se está ejecutando se volverá a ejecutar.

Este tipo de recursividad en el que la llamada recursiva se encuentra al final del procedimiento, suele denominarse recursividad de cola, y serán de este tipo todos los ejemplos que iremos viendo a lo largo de este tomo.

Otro ejemplo similar al anterior es el siguiente:

```
? PARA CUADRADO
> REPITE 4 [ AV 40 GD 90 ]
> CUADRADO
> FIN
```

Si lo ejecutas, en la pantalla de tu ordenador se irán dibujando infinitos cuadrados unos encima de otros.

```
? CUADRADO
```

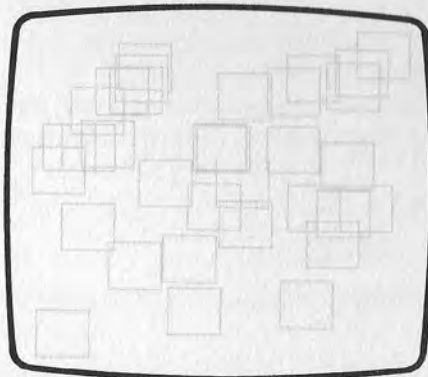
Recuerda que cuando te canses de ver a la Tortuga dibujando cuadrados, puedes detener la ejecución manualmente.

Variando un poco el procedimiento anterior, podemos conseguir que no se dibujen todos los cuadrados en la misma posición.

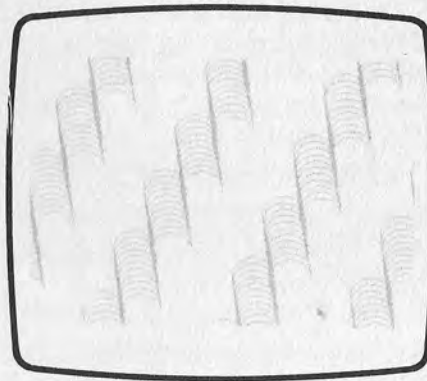
```
? PARA CUADRADO
> BL
> REPITE 4 [ AV 30 GD 90 ]
> SL
> PONY (AZAR 200) - 100
> PONX (AZAR 200) - 100
> CUADRADO
> FIN
```

Lo que hemos hecho ha sido determinar, aleatoriamente, la posición donde se dibujará cada uno de los cuadrados.

```
? CUADRADO
```



Con la recursividad simplificamos enormemente los procedimientos y obtenemos fácilmente pantallas como la siguiente.



```
? PARA CURVA
> PONRUMBO 170
> AV 50
> REPITE 18 [ AV 3 GD 10 ]
> CURVA
> FIN
```

Dependiendo del tiempo que esté ejecutándose el procedimiento CURVA, el dibujo de la pantalla cambiará.

■ Recursividad en procedimientos con una entrada

Hasta ahora hemos visto cómo ejecutar infinitas veces un procedimiento sin entradas desde el mismo procedimiento, hagamos lo mismo pero con procedimientos con una entrada.

El siguiente procedimiento dibuja un círculo cuyo tamaño se determina con un dato de entrada:

```
? PARA CIRCULO :T
> REPITE 36 [ AV :T GD 10 ]
> FIN
```

Ahora cogemos este procedimiento y le añadimos una llamada recursiva:

```
? PARA CIRCULO :T
> MT
> REPITE 36 [ AV :T GD 10 ]
> CIRCULO :T
> FIN
```

Si ejecutamos este último procedimiento.

Utiliza la recursividad y asómbrate.

EXPERIENCIA Y PRACTICAS EN LOGO

? CIRCULO 3

veremos cómo la Tortuga va dibujando círculos del mismo tamaño.

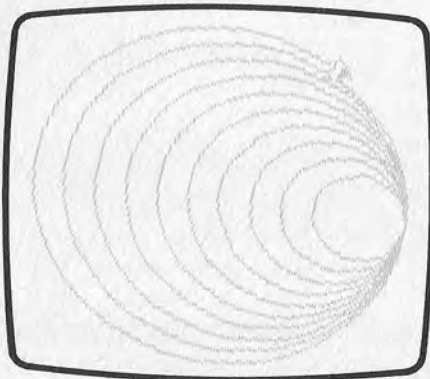
Y los dibuja del mismo tamaño porque en la llamada recursiva no hemos variado el dato de entrada del procedimiento CIRCULO.

Veamos lo que ocurre si incrementamos el dato de entrada:

```
? PARA CIRCULO :T
> MT
> REPITE 36 [ AV :T GD 10 ]
> CIRCULO :T + 1
> FIN
```

Ejecutamos el procedimiento indicando un tamaño inicial igual que en el procedimiento anterior.

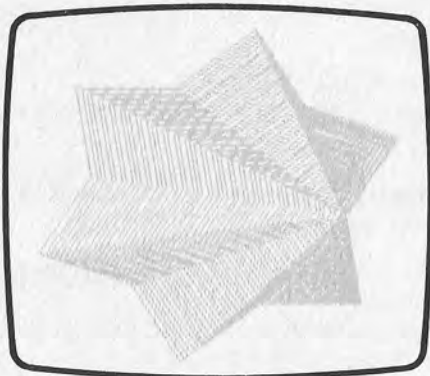
? CIRCULO 3



Hasta que detengamos la ejecución, la Tortuga dibujará círculos de diferentes tamaños.

A continuación ejecutaremos un mismo dibujo a través de dos procesos distintos.

Este será el dibujo a realizar:



— Primero lo realizamos sin utilizar procedimientos recursivos.

El procedimiento ESTRE de una entrada dibuja una estrella cuyo lado está determinado por el valor de la variable L.

```
? PARA ESTRE :L
> PONRUMBO 180
> REPITE T [ AV :L GD 100 AV :L GI 40 ]
> FIN
```

Para ejecutar varias veces el procedimiento ESTRE, definimos este otro procedimiento (ESTRELLA) que inicializa la pantalla y llama 50 veces al procedimiento ESTRE, incrementando (a través de la variable C) cada vez el lado de la estrella en una unidad.

```
? PARA ESTRELLA
> BP
> SL
> PONPOS [ 30 -15 ]
> BL
> HAZ "C 1
> REPITE 50 [ ESTRE :C HAZ "C :C + 1 ]
> FIN
```

— Ahora realizamos el mismo dibujo utilizando un procedimiento recursivo.

```
? PARA ESTRE :L
> SL
> PONPOS [ 30 -15 ]
> BL
> PONRUMBO 180
> REPITE 6 [ AV :L GD 100 AV :L GI 40 ]
> ESTRE :L + 1
> FIN
```

La principal novedad de este procedimiento es que realizamos una llamada recursiva a un procedimiento con una entrada, y que cada vez que realizamos la llamada, variamos el valor del dato que le pasamos al procedimiento.

Cuando ejecutamos el procedimiento ESTRE determinamos el valor inicial de la variable L.

Por ejemplo:

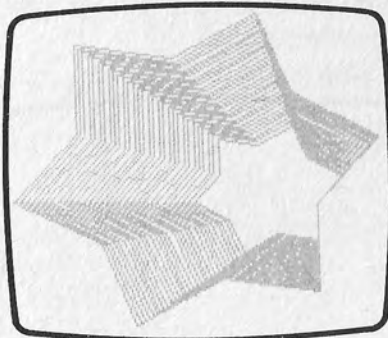
```
? ESTRE 1
```

y obtenemos el dibujo mostrado anteriormente.

Podemos detener manualmente un procedimiento que por sí solo no se detendrá nunca.

También podemos obtener otros dibujos parecidos.

? ESTRE 20



■ Cuadro resumen

— **COMPRUEBA** condición

Determina si la condición que se especifica es **CIERTO** o **FALSO** para utilizarlo posteriormente con **SIC** y **SIF**.

— **SIC** [acción]

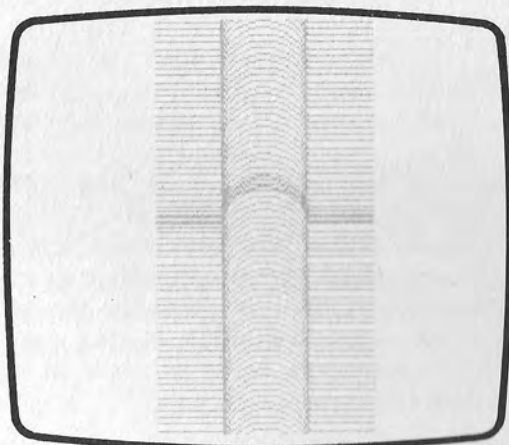
Se ejecuta la acción que se especifica si el resultado del último **COMPRUEBA** ha sido **CIERTO**.

— **SIF** [acción]

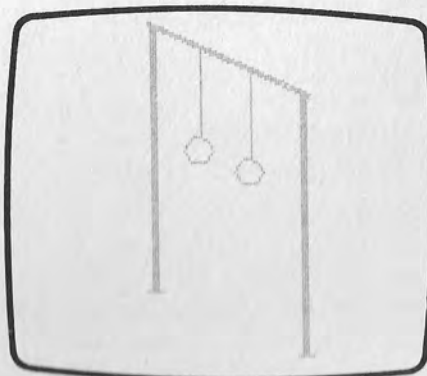
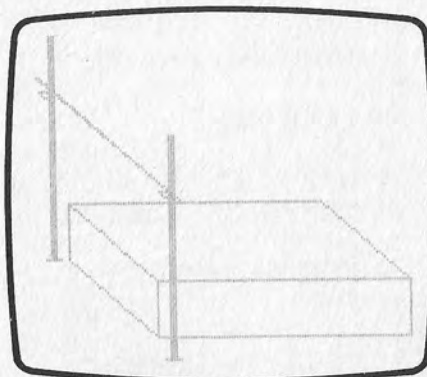
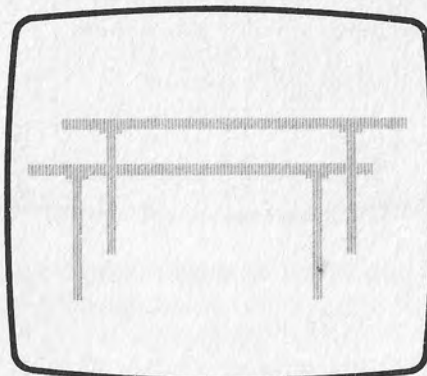
Se ejecuta la acción que se especifica si el resultado del último **COMPRUEBA** ha sido **FALSO**.

■ Ejercicios

1. Define un procedimiento recursivo que realice el siguiente dibujo:

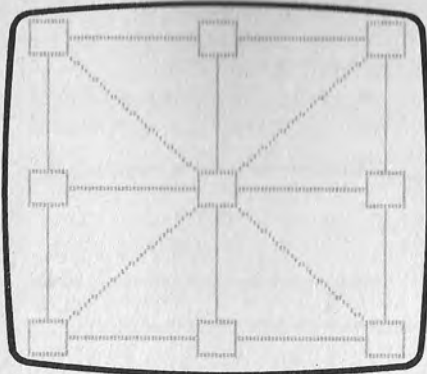


2. Utilizando un procedimiento recursivo, dibuja cubos de distintos tamaños y en la misma posición.
3. Intenta realizar los siguientes dibujos:



4. Intenta definir los procedimientos necesarios para poder jugar a las tres en raya con un amigo tuyo.
Un tablero que puede servir para jugar es:

Una condición también se puede expresar por medio de las órdenes **COMPRUEBA, **SIC** y **SIF**.**



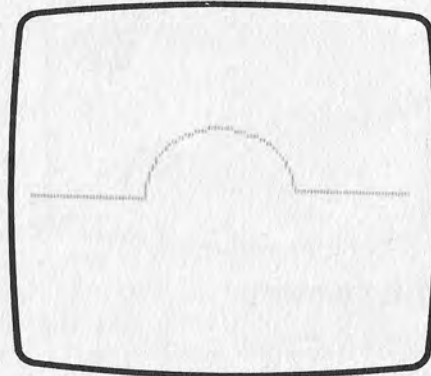
5. ¿Son correctas las órdenes siguientes?

- ? PARA LINEA :C
 - > CENTRO
 - > HAZ "C :C + 10
 - > AV :C
 - > LINEA
 - > FIN
- ? PARA TOR
 - > SL
 - > HAZ "C 20
 - > PONRUMBO AZAR 360
 - > AV :C
 - > TOR
 - > FIN
- ? PARA CUADRADO :L
 - > HAZ "C 20
 - > REPITE 4 [AV :C GD 90]
 - > CUADRADO :L + 3
 - > FIN
- ? HAZ "V 5
 - > COMPRUEBA :V = 5
 - > SIC [AV : V] [ALTO]
- ? PARA A :X
 - > COMPRUEBA :X > 10
 - > SIF [ALTO]
 - > AV :X
 - > FIN
- ? PARA A :X
 - > SIC [AV :X]
 - > SIF [RE :X]
 - > FIN

■ Solución a los ejercicios

1:

El procedimiento CURVA dibuja infinitas veces el siguiente dibujo:



La causa de que este dibujo no se realice siempre en la misma posición es la orden

REPITE 18 [AV 4 GD 10]

pues al finalizar su ejecución la Tortuga no se encuentra en el mismo punto de la coordenada Y que se encontraba al principio.

```
? PARA CURVA
> SL
> PONX -50
> BL
> PONRUMBO 90
> AV 35
> GI 90
> REPITE 18 [ AV 4 GD 10 ]
> GI 90
> AV 35
> CURVA
> FIN
```

2:

CUBO es un procedimiento de una entrada que dibuja un cubo cuyas dimensiones están determinadas por la variable de entrada L.

En el mismo procedimiento y después de dibujar un cubo, una llamada recursiva provoca que el procedimiento vuelva a ejecutarse, incrementando, en la llamada, el valor de la arista del cubo 10 puntos.

Cuando la llamada recursiva se encuentra al final del procedimiento la denominamos recursividad de cola.

```

? PARA CUBO :L
> SL
> CENTRO
> PONX -:L / 2
> PONY -:L / 2
> BL
> REPITE 4 [ AV :L GD 90 ]
> AV :L
> GD 45
> AV :L - ( :L / 3 )
> GD 45
> REPITE 4 [ AV :L GD 90 ]
> AV :L
> GD 90
> AV :L
> GD 45
> AV :L - ( :L / 3 )
> GD 135
> AV :L
> GD 45
> AV :L - ( :L / 3 )
> SL
> PONX -:L / 2
> PONY -:L / 2
> BL
> AV :L - ( :L / 3 )
> CUBO :L + 10
> FIN

```

La llamada recursiva es:

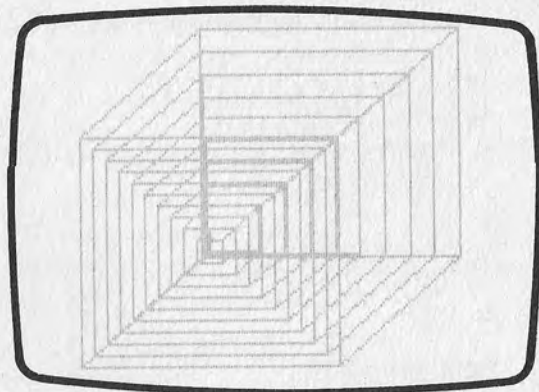
```
CUBO :L + 10
```

Cuando ejecutamos el procedimiento CUBO determinamos el valor inicial de la arista, ese valor se almacena en la variable L, a partir de ese valor se irán dibujando cubos de dimensión L + 10, es decir, el valor actual de la variable L más 10.

La clave del procedimiento es que al realizar la llamada al dato indicado como entrada se vuelve a almacenar en la variable L.

Ejecutamos el procedimiento determinando que la arista del primer cubo que se dibuje sea 10.

```
? CUBO 10
```



3:

En primer lugar, vamos a definir los procedimientos que dibujan las paralelas.

El procedimiento PARALELAS es el que inicializa el proceso y el que realiza las llamadas necesarias a los procedimientos que dibujan las distintas partes de que se componen unas barras paralelas:

```

? PARA PARALELAS
> BP
> OT
> SL
> PONX -50
> BARRA
> SL
> PONPOS [ -40 15 ]
> BARRA
> SL
> PONPOS [ -35 -1 ]
> PATA
> SL
> PONPOS [ 35 -1 ]
> PATA
> SL
> PONPOS [ -25 14 ]
> PATA
> SL
> PONPOS [ 45 14 ]
> PATA
> FIN

```

El procedimiento BARRA dibuja las barras horizontales:

```

? PARA BARRA
> PONRUMBO 90
> REPITE 3 [ BL AV 100 SL RE 100 GI
90 AV 1 GD 90 ]
> FIN

```

Las órdenes SIC y SIF sólo admiten un predicado. COMPRUEBA, sólo tiene acción dentro del procedimiento donde se encuentra.

EXPERIENCIA Y PRACTICAS EN LOGO

y PATA dibuja las patas de sujeción de las barras:

- ? PARA PATA
- > PON RUMBO 180
- > REPITE 3 [BL AV 40 SL RE 40 GD 90 AV 1 GI 90]
- > BL
- > PONRUMBO -90
- > AV 1 RE 6
- > FIN

Para ejecutarlo:

- ? PARALELAS

A continuación definimos los procedimientos que van a dibujar los distintos elementos necesarios para realizar el salto de altura.

ALTURA es el que inicializa el proceso y llama a los procedimientos que dibujan los diferentes elementos:

- ? PARA ALTURA
- > BP
- > OT
- > SL
- > PONY -40
- > COLCHONETA
- > SL
- > PONPOS [5 -50]
- > PATA1
- > SL
- > PONPOS [-42 -5]
- > PATA1
- > SL
- > PONPOS [-47 76]
- > BARRA1
- > FIN

El procedimiento que dibuja la colchoneta es:

- ? PARA COLCHONETA
- > BL
- > REPITE 2 [AV 25 GD 90 AV 100 GD 90]
- > GI 45
- > REPITE 2 [AV 50 GD 45 AV 25 GD 135]
- > GD 45 AV 25 GD 90
- > REPITE 2 [AV 100 GI 135 AV 50 GI 45]
- > FIN

El que dibuja los soportes de la barra que marca la altura es:

- ? PARA PATA1
- > BL
- > PONRUMBO 0
- > REPITE 3 [AV 100 RE 100 GD 90 AV 1 GI 90]
- > GD 90 AV 1
- > RE 6 AV 2
- > GI 90 AV 75
- > GI 90 AV 4
- > GI 135 AV 5
- > FIN

BARRA1 dibuja la barra que marca la altura:

- ? PARA BARRA1
- > REPITE 2 [BL AV 80 SL RE 80 GI 90 AV 1 GD 90]
- > FIN

Para ejecutarlo:

- ? ALTURA

Por último, definimos los procedimientos para dibujar las anillas:

El procedimiento ANILLAS, como en los demás casos, inicializa el dibujo y llama a los diferentes procedimientos:

- ? PARA ANILLAS
- > BP
- > OT
- > SL
- > PONPOS [30 -60]
- > PATA2
- > SL
- > PONPOS [-30 -30]
- > PATA2
- > SL
- > PONPOS [33 57]
- > BARRA2
- > SL
- > PONPOS [10 70]
- > ANILLA
- > SL
- > PONPOS [-10 80]
- > ANILLA
- > FIN

Para dibujar las patas de sujeción y la barra horizontal, definimos:

Utiliza la forma de expresar condicionales más adecuada al proceso que vayas a realizar.

```

? PARA PATA2
> BL
> PONRUMBO 0
> REPITE 3 [ AV 120 RE 120 GD 90 AV
  1 GI 90 ]
> GD 90 AV 1 RE 6
> FIN

? PARA BARRA2
> PONRUMBO -62
> REPITE 3 [ BL AV 73 SL RE 73 GD 90
  AV 1 GI 90 ]
> FIN

```

Y finalmente definimos el procedimiento que dibuja las anillas:

```

? PARA ANILLA
> BL
> PONRUMBO 180
> AV 40 GD 90 REPITE 18 [ AV 2 GI 20
  ]
> FIN

```

Para ejecutarlo:

```

? ANILLAS

```

4:

Para jugar a las tres en raya lo primero que tenemos que tener es un tablero:

```

? PARA TABLERO
> SL
> PONPOS [ -50 -50 ]
> REPITE 3 [ REPITE 3 [ CASILLA SL
  GD 90 AV 45 GI 90 ] SL AV 45 GI 90
  AV 135 GD 90 ]
> SL
> PONPONS [ -40 -40 ]
> PONRUMBO 45
> LINEA1
> SL
> PONPOS [ 40 -40 ]
> PONRUMBO -45
> LINEA1
> SL
> PONPOS [ -40 -45 ]
> PONRUMBO 90
> REPITE 2 [ REPITE 3 [ LINEA2 SL RE
  80 GI 90 AV 45 GD 90 ] SL PONPOS
  [ 45 -40 ] PONRUMBO 0 ]
> FIN

```

El procedimiento TABLERO realiza varias llamadas a diferentes procedimientos. CA-

SILLAS dibuja cuadrados que simulan las nueve casillas que dispone el tablero.

```

? PARA CASILLA
> BL
> REPITE 4 [ AV 10 GD 90 ]
> FIN

```

Para dibujar las líneas inclinadas y las verticales y horizontales definimos respectivamente los procedimientos:

```

? PARA LINEA1
> BL
> AV 50 SL AV 13
> BL
> AV 50
> FIN

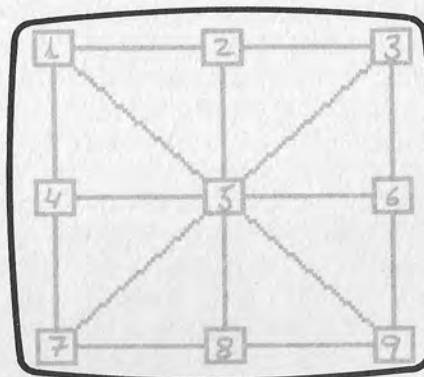
```

```

? PARA LINEA2
> BL
> AV 35
> SL
> AV 10
> BL
> AV 35
> FIN

```

Una vez que tenemos el tablero, vamos a asignar un número a cada una de las casillas:



El procedimiento INICIO inicializa nueve variables a cero. Cada una de estas variables corresponde a una casilla. Si la variable contiene un cero significará que la casilla no ha sido ocupada.

```

? PARA INICIO
> HAZ "A1 0
> HAZ "A2 0
> HAZ "A3 0
> HAZ "A4 0
> HAZ "A5 0

```

SIC y SIF siempre hacen referencia al último COMPRUEBA que se ha realizado.

EXPERIENCIA Y PRACTICAS EN LOGO

- > HAZ "A5 0
- > HAZ "A6 0
- > HAZ "A7 0
- > HAZ "A8 0
- > HAZ "A9 0
- > FIN

El siguiente procedimiento es el que se encarga de leer una tecla pulsada. Realiza dos lecturas del teclado, una para cada jugador y a cada uno le asigna un color diferente.

- ? PARA JUGADA
- > HAZ "T TECLEAR
- > PONCL 15
- > JUEGO
- > HAZ "T LEECAR
- > PONCL 1
- > JUEGO
- > FIN

Después de leer la tecla que se ha pulsado y asignarla a la variable :T, se realiza una llamada al procedimiento JUEGO, que es el que va a investigar sobre cuál es ésta. Dependiendo de cuál sea, se actúa de diferentes formas.

- ? PARA JUEGO
- > SL
- > SI Y (:T = 1) (:A1 = 0) [HAZ "A1 1
PONPOS [-45 45] MARCA ALTO]
- > SI Y (:T = 2) (:A2 = 0) [HAZ "A2 1
PONPOS [0 45] MARCA ALTO]
- > SI Y (:T = 3) (:A3 = 0) [HAZ "A3 1
PONPOS [45 45] MARCA ALTO]
- > SI Y (:T = 4) (:A4 = 0) [HAZ "A4 1
PONPOS [-45 0] MARCA ALTO]
- > SI Y (:T = 5) (:A5 = 0) [HAZ "A5 1
PONPOS [0 0] MARCA ALTO]
- > SI Y (:T = 6) (:A6 = 0) [HAZ "A6 1
PONPOS [45 0] MARCA ALTO]
- > SI Y (:T = 7) (:A7 = 0) [HAZ "A7 1
PONPOS [-45 -45] MARCA ALTO]
- > SI Y (:T = 8) (:A8 = 0) [HAZ "A8 1
PONPOS [0 -45] MARCA ALTO]
- > SI Y (:T = 9) (:A9 = 0) [HAZ "A9 1
PONPOS [45 -45] MARCA ALTO]
- > FIN
- ? PARA MARCA
- > BL
- > RELLENA
- > FIN

En este procedimiento se va preguntando si el valor que contiene la variable :T es igual a uno de los números del 0 al 9. A su vez se pregunta también que si la variable que tiene asignada la casilla correspondiente al número pulsado es igual a cero. Si esta variable es cero, significará que la casilla está libre y, por tanto, la condición se cumple realizándose la acción que se expresa.

La acción que hay que realizar en todos los casos es, en primer lugar, asignar un uno a la variable correspondiente a la casilla elegida, para detectar de alguna forma que ya ha sido utilizada. A continuación se sitúa la Tortuga en el punto que corresponde a la casilla que se ha pulsado. Una vez que está situada, llamamos al procedimiento MARCA que rellena la casilla de un color. Por último, ordenamos con ALTO que se devuelva el control al procedimiento anterior.

Si la variable asignada a cada casilla es uno, significa que ya está ocupada y, por tanto, la condición no se cumple y no se volverá a rellenar aunque sea elegida de nuevo. De esta forma evitamos que la misma casilla sea ocupada más de una vez dentro de la misma jugada.

Una vez que tenemos todos los procedimientos necesarios, los agrupamos en uno general.

- ? PARA 3R
- > BP
- > OT
- > TABLERO
- > INICIO
- > REPITE 5 [JUGADA]
- > FIN

Vamos a ejecutarlo:

- ? 3R

En primer lugar, en la pantalla aparece el tablero dibujado. A continuación el ordenador espera a que una tecla sea pulsada.

Es el turno de que el primer jugador pulse la tecla correspondiente a la casilla que desee pulsar. Debes pulsar una tecla entre el 0 y el 9. Una vez pulsada la casilla correspondiente se rellenará de un color.

A continuación, el segundo jugador deberá realizar el mismo proceso y la casilla ele-

La recursividad es una de las herramientas más potentes que posee el Logo.

gida se rellenará de un color diferente al de su compañero.

Ten cuidado y no intentes ocupar una casilla que ya lo esté, ya que ésta no se rellenará y tu turno se saltará.

Cada jugador dispone de cinco oportunidades y cada vez que se desee volver a jugar, basta con ejecutar el procedimiento 3R.

5:

```
- ? PARA LINEA :C
> CENTRO
> HAZ "C :C + 10
> AV :C
> LINEA
> FIN
```

INCORRECTO

Si ejecutamos este procedimiento, la Tortuga dibujará una línea en la pantalla y luego aparecerá el mensaje de error:

FALTAN DATOS PARA LINEA:

Observa que cuando realizamos la llamada recursiva no determinamos ningún parámetro de entrada, y el procedimiento LINEA es un procedimiento con una entrada.

```
- ? PARA TOR
> SL
> HAZ "C 20
> PONRUMBO AZAR 360
> AV :C
> TOR
> FIN
```

CORRECTO

El procedimiento TOR hace dar tumbos a la Tortuga por toda la pantalla.

```
- ? PARA CUADRADO :I
> HAZ "C 20
> REPITE 4 [ AV :C GD 90 ]
> CUADRADO :L + 3
> FIN
```

CORRECTO

En este procedimiento no hay ningún error para el Logo, pero si analizamos el conjunto de órdenes, vemos que si lo que se pretendía era dibujar cuadrados de tamaño fijo, no es necesario utilizar un procedimiento de una entrada: si, en cambio, se pretendía dibujar cuadrados de distintos tamaños, no tiene sentido incluir la segunda orden del procedimiento y en la tercera orden debería aparecer la variable L en lugar de la C.

```
- ? HAZ "V 5
? COMPRUEBA :V = 5
? SIC [ AV :V ] [ ALTO ]
```

INCORRECTO

En la orden SIC sólo se puede expresar una acción a tomar.

```
- ? PARA A :X
> COMPRUEBA :X > 10
> SIF [ ALTO ]
> AV :X
> FIN
```

CORRECTO

Al ejecutar el procedimiento introducimos el valor a la variable X. Comprobamos si el valor introducido es mayor que diez. Si lo es, no se ejecuta la línea donde aparece el SIF (ya que la condición devuelve el resultado CIERTO) y pasa a la línea siguiente. La Tortuga avanza los pasos que le indica X y acabará el procedimiento. Si el valor de X es menor de diez, el procedimiento se detendrá, ya que la condición devuelve el resultado FALSO y se ejecutará la acción que acompaña al SIF.

```
- ? PARA A :X
> SIC [ AV :X ]
> SIF [ RE :X ]
> FIN
```

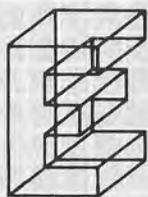
INCORRECTO

Para utilizar la orden SIC y SIF es necesario que previamente exista un COMPRUEBA.

Con el Logo podemos crear juegos y divertirnos con ellos.

SIC y SIF pueden no ir seguidas.

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS



N el tomo anterior vimos dos de los tres tipos de tablas que había. El primero de ellos, que estuvo incluido en el programa GUSANO LOCO, lo explicamos, pero el segundo no, por falta de espacio. Lo primero que veremos en este nuevo tomo será el funcionamiento de dicho programa. Después veremos otra rutina que nos mostrará la tercera y más perfeccionada forma de realizar tablas de RECORDS.

Comencemos con el programa 2 del tomo anterior; su explicación línea a línea es la siguiente:

Como puede apreciarse a primera vista, las líneas de esta rutina van de cinco en cinco en vez de ir de diez en diez. Esto es así porque si hubiésemos reenumerado el programa de diez en diez, al intentar incluirlo en el programa GUSANO LOCO, los números de línea hubiesen sido mayores de los permitidos al final de la rutina y se hubiesen comido la parte del programa que nos avisaba del final del juego.

Líneas 2475 y 2480. Se realiza un bucle de retardo. Este bucle, al igual que en el programa anterior, no sirve para nada pero como hemos sacado esta rutina del programa GUSANO LOCO y como cabe la posibilidad de incluir esta nueva rutina en dicho programa, se han incluido estas dos líneas para guardar la compatibilidad entre la rutina y el programa. Por supuesto, si el usuario quiere utilizar esta rutina en sus propios programas, puede quitar libremente estas dos líneas.

El grupo de líneas que vienen a continuación y que van desde la 2485 a la 2520 se encargan de dibujarnos en pantalla el rectán-

gulo dentro del cual aparecerá la tabla de RECORDS. Esta parte del programa ya ha sido explicada en la rutina anterior. Lo único que tienen diferente las dos rutinas son los números de línea, pero en este caso no es importante.

Entre las líneas 2525 y 2550 se comprueba que la puntuación del jugador merece estar entre las diez mejores; la incluimos en la tabla en caso de que así sea y ordenamos la tabla de mayor a menor. Esta parte del programa también ha sido explicada.

Las líneas 2555, 2560 y 2565 nos dicen en qué posición de la tabla ha quedado almacenado el nuevo RECORD (en el caso de que lo hubiese).

Línea 2570. Si no ha habido nuevo RECORD, nos vamos a la línea 2600 para imprimir la tabla de ganadores.

Línea 2575. Hacemos que una cadena de tres asteriscos (***) se almacene en A\$(PP). Esta es el elemento del vector A\$() donde almacenaremos el nombre del jugador que acaba de conseguir un nuevo RECORD.

Líneas 2580 a 2595. Imprimimos la tabla entera de RECORDS, incluido el del jugador (si lo ha conseguido). En este caso, cada nombre está compuesto de sólo tres letras. Cuando aparezca la puntuación del jugador, en el lugar del nombre aparecerán tres asteriscos. En esta posición será donde el jugador tendrá que introducir las iniciales de su nombre.

Línea 2600. Si no ha habido RECORD, como no tenemos que preguntar el nombre del jugador, nos saltamos las líneas encargadas de hacerlo y nos vamos a la línea 2715 para imprimir el mensaje de PULSA UNA TECLA.

Línea 2605. Colocamos el cursor al principio de los tres caracteres que formarán el nombre del jugador.

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

Línea 2610. Borrarnos los tres asteriscos que habíamos almacenado en A\$(PP) para empezar con la recogida del nombre sin tener nada almacenado en esta variable.

Línea 2615. Asignamos el valor dos (2) a la variable numérica C. Esta variable será la que almacene el carácter que se está imprimiendo en la pantalla en un cierto momento. Veremos más adelante cómo funciona y cómo la utilizamos.

Línea 2620. Asignamos a la variable X el valor cero (0). Esta variable nos servirá para comprobar que el usuario ha pulsado la tecla de disparo para elegir la letra que está viendo.

Línea 2625. Almacenamos en la variable alfanumérica F\$ todos los caracteres que el usuario podrá utilizar para introducir sus iniciales.

Línea 2630. Como el nombre del jugador que ha conseguido un nuevo RECORD sólo puede tener tres caracteres como máximo, comenzamos un bucle de tres vueltas cuya variable índice será la I.

Línea 2635. Colocamos el cursor de la letra de su nombre que le corresponda al jugador elegir para escribirlo.

Línea 2640. E imprimimos la letra que en ese momento esté disponible. Dicha letra será una de las almacenadas en la variable Z\$ y su posición dentro de la cadena vendrá dada por el valor de C.

Línea 2645. Miramos a ver si el jugador ha pulsado alguna tecla. En caso negativo volvemos a preguntar hasta que lo haga.

Línea 2650. Si la tecla pulsada es una P (mayúscula o minúscula), entonces hacemos que C se incremente en uno para, más adelante, imprimir la siguiente letra de la lista.

Línea 2655. Si lo que puso fue una «O» o una «o», entonces decrementamos en uno el valor de C para imprimir la letra anterior.

Línea 2660. Si pulsó la tecla M mayúscula o la M minúscula entonces significa que el usuario quiere fijar la letra que está visualizando en ese momento. En ese caso ponemos la variable X con valor menos uno (-1). Este valor nos dirá más adelante que dicha letra ha sido elegida.

Línea 2665. Si no se pulsó ninguna de las letras que acabamos de ver, entonces nos volvemos a la línea 2645 para seguir preguntando hasta que se pulse una de las permitidas.

Línea 2670. Colocamos el cursor en la posición del nombre que corresponda y que

tiene relación directa con la letra que estamos editando y, por tanto, con el valor de la variable índice del bucle.

Línea 2675. Si la tecla que pulsó el jugador fue la de fuego, entonces nos vamos a la línea 2700 para procesar el almacenamiento de dicha letra.

Línea 2680. Si el valor de C es mayor que el de letras disponibles. Si el valor de C ha excedido del número de letras que tenemos en la lista, entonces hacemos que tome valor uno (1).

Línea 2685. Si, por el contrario, el valor de C es menor que cero, como nos estamos saliendo de la lista, pero por el otro extremo, hacemos que C sea igual al número máximo de letras en la lista para que, así, apunte a la última letra de ésta.

Línea 2690. Imprimimos la letra que acaba de ser seleccionada.

Línea 2695. Y nos volvemos a la línea 2645 en espera de una nueva pulsación desde el teclado.

Línea 2700. Aparecemos en esta línea cuando el usuario ha pulsado la tecla fuego. En ese caso tenemos que almacenar la letra que se estaba visualizando. La posición de dicha letra en la cadena alfanumérica Z\$ nos la da la variable C. Por ello, concatenamos el valor que tenga la variable A\$(PP) con la letra que se encuentre en dicha posición.

Línea 2705. Hacemos que X tenga valor cero de nuevo.

Línea 2710. Y terminamos el bucle para continuar con la siguiente letra.

Las líneas que vienen a continuación son iguales que las que vimos en el tomo anterior al explicar el programa número uno.

A continuación veremos un programa que nos servirá como referencia para conocer y aprender el tercer tipo de tabla de RECORDS que podemos realizar. Este nuevo tipo se caracteriza porque toda las letras disponibles están permanentemente en pantalla y nos movemos, con un cursor, de una a otra antes de elegir cuál es la que deseamos.

Este programa sólo admite nombres de tres letras, pero puede ser ampliado sin mucha dificultad para que admita nombres de cualquier longitud.

Otra particularidad de esta rutina es que podemos borrar el nombre, si nos hemos equivocado, y podemos poner fin a la impresión de nuestro nombre en cuanto queramos. Esto lo podemos realizar moviéndonos a la «E»

minúscula que se encuentra en la última fila de caracteres (en el caso de terminar) o a la

«B» minúscula que se encuentra al lado de ésta (en el caso de que queramos borrar).

```
2450 REM
2455 REM *****
2460 REM * TABLA DE GANADORES *
2465 REM *****
2470 REM
2475 FOR I=1 TO 1000
2480 NEXT I
2485 CLS
2490 PRINT "PUNTUACIONES DE HOY"
2495 PRINT
2500 PRINT "-----"
2505 FOR I=4 TO 17
2510     PRINT "!";TAB(20);"!"
2515 NEXT I
2520 PRINT "-----"
2525 LET Z$="ABCDEFGHIJKLMNOPQRSTUVWXYZ ,.:!?$%/&*()-=+<>[]eb"
2530 LET P=4
2535 LOCATE P,25
2540 FOR I=1 TO LEN(Z$) STEP 10
2545     FOR J=I TO I+9
2550         PRINT MID$(Z$,J,1);
2555     NEXT J
2560     LET P=P+2
2565     LOCATE P,25
2570 NEXT I
2575 IF SC>=S(10) THEN LET S(10)=SC:LET A$(10)=CHR$(254)
2580 FOR I=1 TO 10
2585     FOR J=1 TO I
2590         IF S(J)<S(I) THEN LET A=S(J):LET S(J)=S(I):LET S(I)=A:LET B$=A$(J):LET
2595         T A$(J)=A$(I):LET A$(I)=B$
2600 NEXT J
2605 FOR I=1 TO 10
2610     IF A$(I)=CHR$(254) THEN LET PP=I
2615 NEXT I
2620 IF PP=0 THEN GOTO 2630
2625 LET A$(PP)="***"
2630 FOR I=1 TO 10
2635     LOCATE 5+I,3
2640     PRINT A$(I);" ....";S(I)
2645 NEXT I
2650 IF PP=0 THEN GOTO 2825
2655 LET P=5
2660 LET X=25
2665 LET A$(PP)=" "
2670 FOR I=1 TO 3
2675     LOCATE P,X
2680     PRINT "*"
2685     LET B$=INKEY$
2690     IF B$="" THEN GOTO 2685
2695     IF B$="P" OR B$="p" THEN LET C=1:F=0:GOTO 2725
2700     IF B$="O" OR B$="o" THEN LET C=-1:LET F=0:GOTO 2725
2705     IF B$="Q" OR B$="q" THEN LET C=0:LET F=-1:GOTO 2725
2710     IF B$="A" OR B$="a" THEN LET C=0:LET F=1:GOTO 2725
2715     IF B$="M" OR B$="m" THEN LET C=0:LET F=0:GOTO 2725
```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```

2730 GOTO 2685
2735 LOCATE P,X
2730 PRINT " "
2735 IF C=0 AND F=0 THEN GOTO 2785
2740 LET P=P+2*F
2745 LET X=X+C
2750 LET C=0
2755 LET F=0
2760 IF P>13 THEN P=5
2765 IF P<5 THEN LET P=13
2770 IF X>34 THEN LET X=25:LET P=P+2:GOTO 2760
2775 IF X<25 THEN LET X=34:LET P=P-2:GOTO 2760
2780 GOTO 2675
2785 LET N=((P-7)/2+1)*10+X-24
2790 IF MID$(Z$,N,1)="e" THEN LET A$(PP)=LEFT$(A$(PP)+" ",3):LET I=3:GOTO 2
810
2795 IF MID$(Z$,N,1)="b" AND I>1 THEN LET I=I-2:LET A$(PP)=LEFT$(A$(PP),I):LO
CATE 5+PP,3:PRINT "***":GOTO 2810
2800 IF MID$(Z$,N,1)="b" THEN GOTO 2675
2805 LET A$(PP)=A$(PP)+MID$(Z$,N,1)
2810 LOCATE 5+PP,3
2815 PRINT A$(PP)
2820 NEXT I
2825 LOCATE 20,1
2830 PRINT "<< PULSA UNA TECLA >>"
2835 LET B$=INKEY$
2840 IF B$="" THEN GOTO 2835
2845 LOCATE 20,1
2850 GOTO 2330

```



Como siempre, las modificaciones que hay que realizar para que el programa funcione en ordenadores distintos del IBM y compatibles son las siguientes:

COMMODORE:

```

2845 PRINT CHR$(147)
2510 PRINT "!";TAB(18);"!";
2535 POKE 214,P:POKE 211,24
2565 POKE 214,P:POKE 211,24
2635 POKE 214,4+I:POKE 211,2
2660 LET X=24
2675 POKE 214,P:POKE 211,X
2685 GET B$
2725 POKE 214,P:POKE 211,X
2770 IF X>33 THEN LET X=25:LET P=P+2:GOTO 2760

```

```

2775 IF X<24 THEN LET X=33:LET P=P-2:GOTO 2760
2795 IF MID$(Z$,N,1)="b" AND I>1 THEN LET I=I-2:LET
A$(PP)=LEFT$(A$(PP),I):POKE 214,4+PP:POKE
211,2:PRINT "****":GOTO 2810
2810 POKE 214,4+PP:POKE 211,2
2825 POKE 214,19:POKE 211,0
2835 GET B$
2845 POKE 214,19:POKE 211,0

```

SPECTRUM:

```

2510 PRINT "!";TAB(17);"!";
2835 PRINT AT P,20;
2550 PRINT Z$(J);
2565 PRINT AT P,20;
2635 PRINT AT 4+I,2;
2660 LET X=20
2675 PRINT AT P,X;
2725 PRINT AT P,X;
2770 IF X>29 THEN LET X=20:LET P=P+2:GOTO 2760
2775 IF X<20 THEN LET X=29:LET P=P-2:GOTO 2760
2790 IF Z$(N)="e" THEN LET A$(PP)=(A$(PP)+" ") (TO
3):LET I=3:GOTO 2810
2795 IF Z$(N)="b" AND I>1 THEN LET I=I-2:LET
A$(PP)=A$(PP, TO I):PRINT AT 4+PP,2;****:GOTO
2810
2800 IF Z$(N)="b" THEN GOTO 2675
2805 LET A$(PP)=A$(PP)+Z$(N)
2810 PRINT AT 4+PP,2;
2825 PRINT AT 19,0;
2845 PRINT AT 19,0;

```

MSX y AMSTRAD:

```

2535 LOCATE 25,P
2565 LOCATE 25,P
2635 LOCATE 3,5+I
2675 LOCATE X,P
2725 LOCATE X,P
2795 IF MID$(Z$,N,1)="b" AND I>1 THEN LET I=I-2:LET
      A$(PP)=LEFT$(A$(PP),I):LOCATE 3,5+PP:PRINT
      *****GOTO 2810
2810 LOCATE 3,5+PP
2825 LOCATE 1,20
2845 LOCATE 1,20

```

Este nuevo programa es muy parecido a los dos anteriores. También puede ser unido al programa GUSANO LOCO para poder ver mejor cómo funciona. Por supuesto, que el programa 3 que se dio en el tomo anterior nos puede servir para ver cómo funciona. A continuación volvemos a transcribirlo para comodidad de los lectores.

```

100 REM *****
110 REM * PROGRAMA DEMO PARA VER *
120 REM * EL FUNCIONAMIENTO DE *
130 REM * LA RUTINA DE RECORDS. *
140 REM *****
150 REM
160 CLS
170 DIM A$(10)
180 DIM S(10)
190 LET PP=0
200 FOR I=1 TO 10
210   READ A$(I),S(I)
220 NEXT I
230 INPUT "QUE PUNTUACION QUIERES TENER ";SC
240 GOSUB 2450
250 CLS
260 PRINT "FIN DE LA DEMOSTRACION"
270 END
280 REM
290 REM *****
300 REM * DATAS CON NOMBRES Y PUNTUACIONES FICTICIAS *
310 REM *****
320 REM
330 DATA "FMG",12455
340 DATA "JYH",11022
350 DATA "ABA",10344
360 DATA "RIP",9800
370 DATA "PUS",9780
380 DATA "RES",9201
390 DATA " A ",7200
400 DATA "Z Z",6543
410 DATA "ZIP",3403
420 DATA "ZAP",2195

```



Las únicas modificaciones que hay que realizar para que funcione en el SPECTRUM y en el COMMODORE, aunque ya aparecieron en el tomo anterior, son las siguientes:

COMMODORE:

```

160 PRINT CHR$(147)
250 PRINT CHR$(147)

```

SPECTRUM:

```

170 DIM A$(10,3)

```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

Veamos ahora cómo funciona la parte central de esta última rutina. El resto de las líneas no van a ser explicadas por su parecido a las de los dos programas anteriores. Así, pues, empezaremos a ver el programa a partir de la línea 2655:

Línea 2655. Asignamos a la variable P el valor 5. Esta variable nos indicará en qué línea, de las cinco que disponemos, se encuentra el cursor.

Línea 2660. Almacenamos en X el valor de la columna donde empiezan las filas de letras.

Línea 2665. Borramos el contenido de la variable A\$(PP).

Línea 2670. Comenzamos un bucle dentro del cual vamos a leer las tres letras que el usuario quiere que definan su nombre.

Línea 2675. Colocamos el cursor debajo del primer carácter y en la primera de las cinco líneas que contienen los caracteres que podemos definir.

Línea 2680. Imprimimos un asterisco (*) para indicar en qué posición nos encontramos.

Línea 2685. Miramos si el usuario ha pulsado alguna tecla.

Línea 2690. Si el usuario no ha pulsado ninguna tecla, entonces volvemos a preguntar.

Líneas 2695 a 2715. Comprobamos que el usuario ha pulsado alguna de las teclas permitidas. Según la tecla pulsada, variamos el valor de unas u otras variables. La variable C nos dice si hay que ir a la columna siguiente (1) o a la anterior (-1). La variable F nos indica si tenemos que ir a la línea siguiente (1) o a la anterior (-1). Si hemos pulsado la tecla de disparo, ambas variables valen cero.

Línea 2720. Si no pulsó ninguna de las teclas autorizadas, nos volvemos a la línea 2685.

Línea 2725. Colocamos el cursor en la posición donde estaba el asterisco.

Línea 2730. Y lo borramos.

Línea 2735. Preguntamos si pulsó la tecla de disparo. En caso afirmativo, nos vamos a la línea 2785.

Línea 2740. Calculamos la nueva línea a la que tenemos que ir.

Línea 2745. Y la nueva columna.

Línea 2750. Ponemos la variable C a cero (0).

Línea 2755. Y la variable F también a cero.

Línea 2760. Si hemos llegado a la última línea, nos vamos a la primera.

Línea 2765. Si hemos llegado a la última línea, nos vamos a la última.

Línea 2770. Si hemos llegado al último carácter de una línea, nos vamos al principio de la siguiente.

Línea 2775. Si hemos llegado al principio de una línea, nos vamos al final de la anterior.

Línea 2780. Nos volvemos a la línea 2675 para dibujar el asterisco en su nueva posición.

Línea 2785. Calculamos la posición dentro de Z\$ del carácter sobre el que se encuentra el cursor-asterisco.

Línea 2790. Si éste es el carácter «e», hemos terminado y nos vamos a la línea 2810.

Línea 2795. Si este carácter es la «b» y ya habíamos introducido algún carácter, nos vamos a la línea 2810.

Línea 2800. Si el carácter introducido era una «b», pero no habíamos introducido ninguna letra anteriormente, nos vamos a la línea 2675.

Línea 2805. Almacenamos la letra elegida.

Línea 2810. Colocamos el cursor en la posición donde corresponda.

Línea 2815. E imprimimos la tecla almacenada.

Línea 2820. Aquí termina el bucle.

TRUCOS Y RUTINAS BASICAS



OMO ya se encuentra cerca la hora en que tendremos que crear nuestro primer juego en serio dentro de la sección de MANEJO DE SPRITES Y ELEMENTOS GRAFICOS, vamos a ver en éste y en tomos sucesivos

unos programas que nos permitirán definir nuestros propios caracteres gráficos. Veremos un programa para el SPECTRUM, otro para el COMMODORE, otro para el MSX y otro para el AMSTRAD. Para el IBM no incluiremos ninguno, ya que con este ordenador no se pueden definir ni caracteres ni SPRITES. Aun así, más adelante, cuando lleguemos a los capítulos que hablan de alta resolución, veremos cómo crear nuestras propias figuras en el IBM.

Los programas que veremos en esta serie de tomos serán los siguientes:

— Un diseñador de SPRITES para el MSX.

— Un diseñador de SPRITES y caracteres para el COMMODORE.

— Un diseñador de caracteres para el AMSTRAD.

No explicaremos el funcionamiento de cada uno de estos programas debido a su extensión. Se han pensado estos programas como herramienta de trabajo y no como ejemplo para explicar una serie de conceptos o de formas de realizar una cierta idea o función con nuestro ordenador. Por ello, se puede decir que los programas que van a ir apareciendo pueden ser considerados como profesionales, debido a su utilidad real y a su funcionamiento.

El primer programa que vamos a ver corresponde a la versión para SPECTRUM del diseñador de caracteres. Ni qué decir tiene que estos programas no son versiones unos de otros, sino que cada uno de ellos ha sido realizado específicamente para cada ordenador para poder sacar el mayor partido posible a cada uno de ellos.

```
100 INK 0: PAPER 7: BORDER 7: CLS
110 DEF FN l(A)=A-256*INT (A/256)
120 DEF FN h(A)=INT (A/256)
130 POKE 23658,8
140 REM1987 AGB
150 REM CREADOR DE UDG'S
160 REM INICIALIZACION
170 CLEAR 63687
180 LET DIRBAS=63688
190 LET BANCO=0
200 PRINT AT 11,2; FLASH 1;"POR FAVOR, ESPERE MEDIO MINUTO"
210 FOR g=0 TO 9
220 LET t=g*168+dirbas
230 FOR i=0 TO 167
240 POKE T+i,PEEK (USR "a"+i)
250 NEXT i
260 NEXT g
270 GO SUB 3090
280 RESTORE 330
290 FOR I=USR "A" TO USR "B"+7
```

TRUCOS Y RUTINAS BASICAS

```

300 READ A
310 POKE I,A
320 NEXT I
330 DATA 0,126,66,66,66,66,126,0,0,126,126,126,126,126,0
340 CLS
350 LET x1=0
360 LET y1=0
370 LET l=32
380 LET m$="Definicion"
390 GO SUB 2830
400 LET x1=16
410 LET l=13
420 LET y1=2
430 LET m$="Pizarra"
440 GO SUB 2830
450 LET x1=3
460 LET l=9
470 LET m$="Grafico"
480 GO SUB 2830
490 LET y1=16
500 LET x1=0
510 LET l=32
520 LET m$="Opciones"
530 GO SUB 2830
540 DIM g$(8,8)
550 DIM P$(8,9)
560 FOR i=1 TO 8
570 LET g$(i)="AAAAAAAA"
580 NEXT i
590 GO SUB 3130
600 GO SUB 2980
610 PLOT 143,144
620 DRAW 73,0
630 DRAW 0,-65
640 DRAW -73,0
650 DRAW 0,65
660 GO SUB 3180
670 GO SUB 3250
680 LET c$="A"
690 LET x=3
700 LET y=4
710 GO SUB 3310
720 REM COMIENZO DEL BUCLE PRICIPAL
730 PRINT AT Y,X;" "
740 PAUSE 12: IF INKEY$<>" " THEN PRINT AT Y,X;G$(Y-3,X-2): GO TO 770
750 PRINT AT Y,X;G$(Y-3,X-2)
760 PAUSE 12
770 LET A$=INKEY$
780 IF A$<"9" AND A$>"0" THEN GO TO 840
790 LET X=X+(A$="P" AND X<10)-(A$="O" AND X>3)
800 LET Y=Y+(A$="A" AND Y<11)-(A$="Q" AND Y>4)
810 IF A$<>" " THEN GO TO 730
820 LET G$(Y-3,X-2)=("A" AND G$(Y-3,X-2)="A")+("B" AND G$(Y-3,X-2)="B")
830 GO TO 730
840 REM OPCIONES
850 BEEP .1,24
860 IF A$="1" THEN GO SUB 970: GO TO 940
870 IF A$="2" THEN GO SUB 1050: GO TO 940
880 IF A$="3" THEN GO SUB 1210: GO TO 940
890 IF A$="4" THEN GO SUB 1270: GO TO 940
900 IF A$="5" THEN GO SUB 1640: GO TO 940
910 IF A$="6" THEN GO SUB 2090: GO TO 940
920 IF A$="7" THEN GO SUB 2650: GO TO 940
930 IF A$="8" THEN GO SUB 2710
940 INPUT "": PRINT #1;" O,P,Q,A-Mover 1-8 Opciones"
950 BEEP .1,24

```

```

960 GO TO 730
970 REM CAMBIO DE BANCO
980 INPUT "Seleccione banco (1-9) ";b
990 IF b>9 OR b<0 THEN GO TO 3470
1000 LET BANCO=B
1010 LET DIRBAS=63688+168*BANCO
1020 GO SUB 2980
1030 GO SUB 3180
1040 RETURN
1050 REM ASIGNA CHARACTER
1060 INPUT "Caracter (A-U) ? "; LINE b$
1070 IF b$>"U" OR b$<"A" THEN GO TO 3470
1080 LET c$=b$
1090 GO SUB 3130
1100 FOR g=1 TO 8
1110 LET t=0
1120 LET T2=128
1130 FOR i=1 TO 8
1140 LET t=t+(T2 AND g$(g,I)="B")
1150 LET T2=T2/2
1160 NEXT i
1170 POKE USR C$+G-1,T
1180 NEXT g
1190 GO SUB 2980
1200 RETURN
1210 REM COPIA DE UN CHARACTER A LA PARRILLA
1220 INPUT "Caracter (A-U) ? "; LINE b$
1230 IF b$>"U" OR b$<"A" THEN GO TO 3470
1240 LET C$=B$
1250 GO SUB 3310
1260 RETURN
1270 REM SAVE/LOAD
1280 INPUT "Save o Load ? (S/L) "; LINE b$
1290 IF b$<>"S" AND b$<>"L" THEN GO TO 3470
1300 LET x1=0
1310 LET y1=16
1320 LET m$="Save"
1330 LET L=32
1340 IF b$="L" THEN LET m$="Load"
1350 GO SUB 3510
1360 GO SUB 2830
1370 INPUT "Nombre (Max.10 caracteres)"; LINE n$
1380 IF LEN n$>10 THEN GO TO 1550
1390 IF b$="L" THEN GO TO 1490
1400 REM SAVE
1410 IF N$="" THEN GO TO 1550
1420 SAVE N$CODE DIRBAS,168
1430 INPUT "Desea verificar ? (S/N) "; LINE b$
1440 IF b$="N" THEN GO TO 1570
1450 PRINT AT 17,0;"Ponga en marcha el cassette"
1460 PRINT AT 17,0;
1470 VERIFY n$CODE dirbas,168
1480 GO TO 1570
1490 REM LOAD
1500 PRINT AT 17,0;"Ponga en marcha el cassette"
1510 LOAD n$CODE dirbas,168
1520 GO SUB 2980
1530 GO SUB 3180
1540 GO TO 1570
1550 REM ERROR EN EL NOMBRE
1560 GO SUB 3470
1570 GO SUB 3510
1580 LET X1=0
1590 LET Y1=16
1600 LET M$="Opciones"
1610 GO SUB 2830

```

TRUCOS Y RUTINAS BASICAS

```

1620 GO SUB 3250
1630 RETURN
1640 REM EDICION DE LA PIZARRA
1650 GO SUB 3510
1660 LET X1=0
1670 LET Y1=16
1680 LET M$="Edicion de pizarra"
1690 LET L=32
1700 GO SUB 2830
1710 PRINT AT 18,0;"5,6,7,8-Movimiento del cursor.""SPACE-Borra el caracter."
1720 PRINT AT 20,0;"A-U-Poner caracter en la pizarra"
1730 PRINT AT 21,0;"ENTER-Finaliza edicion."
1740 LET X2=18
1750 LET Y2=4
1760 REM BUCLE DE LA EDICION DE PIZARRA
1770 FOR I=1 TO 8
1780 FOR G=1 TO 9
1790 LET B$=P$(I,G)
1800 IF B$<>" " THEN LET B$=CHR$(CODE B$-79)
1810 LET P$(I,G)=B$
1820 NEXT G
1830 NEXT I
1840 GO SUB 3180
1850 BEEP .1,24
1860 PRINT AT Y2,X2; PAPER 2;" "
1870 PAUSE 10
1880 IF INKEY$<>" " THEN PRINT AT Y2,X2;P$(Y2-3,X2-17): GO TO 1910
1890 PRINT AT Y2,X2;P$(Y2-3,X2-17)
1900 PAUSE 10
1910 LET A$=INKEY$
1920 IF A$=CHR$ 13 THEN GO TO 1980
1930 LET X2=X2+(A$="8" AND X2<26)-(A$="5" AND X2>18)
1940 LET Y2=Y2+(A$="6" AND Y2<11)-(A$="7" AND Y2>4)
1950 IF A$<>" " AND (A$<"A" OR A$>"U") THEN GO TO 1860
1960 LET P$(Y2-3,X2-17)=A$
1970 GO TO 1860
1980 REM SALIDA DE LA OPCION 5
1990 BEEP .1,24
2000 FOR I=1 TO 8
2010 FOR G=1 TO 9
2020 LET B$=P$(I,G)
2030 IF B$<>" " THEN LET B$=CHR$(CODE B$+79)
2040 LET P$(I,G)=B$
2050 NEXT G
2060 NEXT I
2070 GO SUB 3180
2080 GO TO 1570
2090 REM OPERACIONES EXTRA
2100 GO SUB 3510
2110 LET X1=0
2120 LET Y1=16
2130 LET L=32
2140 LET M$="Opciones extra"
2150 GO SUB 2830
2160 PRINT AT 18,0;"1-Volver al menu4-Espejo horiz.""2-Invertir.""5-Rotar izq."
2170 PRINT AT 20,0;"3-Espejo vert.""6-Rotar derecha."
2180 LET a$=INKEY$
2190 IF A$>"6" OR A$<"1" THEN GO TO 2180
2200 BEEP .1,24
2210 IF a$="1" THEN GO TO 1570
2220 IF a$="2" THEN GO SUB 2300
2230 IF a$="3" THEN GO SUB 2380
2240 IF a$="4" THEN GO SUB 2450
2250 IF a$="5" THEN GO SUB 2540
2260 IF a$="6" THEN GO SUB 2590

```

```

2270 BEEP .1,24
2280 GO SUB 2920
2290 GO TO 2180
2300 REM INVERSION DE CARACTER
2310 FOR I=1 TO 8
2320 FOR G=1 TO 8
2330 IF G$(I,G)="A" THEN LET G$(I,G)="B": GO TO 2350
2340 LET G$(I,G)="A"
2350 NEXT G
2360 NEXT I
2370 RETURN
2380 REM ESPEJO VERTICAL
2390 FOR I=1 TO 4
2400 LET B$=G$(I)
2410 LET G$(I)=G$(9-I)
2420 LET G$(9-I)=B$
2430 NEXT I
2440 RETURN
2450 REM ESPEJO HORIZONTAL
2460 FOR G=1 TO 8
2470 FOR I=1 TO 4
2480 LET B$=G$(G,I)
2490 LET G$(G,I)=G$(G,9-I)
2500 LET G$(G,9-I)=B$
2510 NEXT I
2520 NEXT G
2530 RETURN
2540 REM ROTAR IZQUIERDA
2550 FOR I=1 TO 8
2560 LET G$(I)=G$(I,2 TO )+G$(I,1)
2570 NEXT I
2580 RETURN
2590 REM ROTAR DERECHA
2600 FOR I=1 TO 8
2610 LET G$(I)=G$(I,8)+G$(I, TO 7)
2620 NEXT I
2630 RETURN
2640 STOP
2650 REM BORRADO DE LA PARRILLA
2660 FOR I=1 TO 8
2670 LET G$(I)="AAAAAAAA"
2680 NEXT I
2690 GO SUB 2920
2700 RETURN
2710 REM VUELTA A BASIC
2720 GO SUB 3510
2730 LET X1=0
2740 LET Y1=16
2750 LET L=32
2760 LET M$="Programa abandonado"
2770 GO SUB 2830
2780 PRINT AT 18,0;"Para volver al programa,teclear"
2790 STOP
2800 GO TO 1570
2810 LET X=5: LET Y=5
2820 RETURN
2830 REM coloca el encabezamiento
2840 FOR i=175-y1*8 TO 175-y1*8-7 STEP -2
2850 PLOT x1*8,i
2860 DRAW l*8-1,0
2870 NEXT i
2880 INVERSE 1
2890 PRINT AT y1,x1+INT ((1-LEN m$)/2);m$
2900 INVERSE 0
2910 RETURN
2920 REM escribe parrilla

```

CONTINUE"

TRUCOS Y RUTINAS BASICAS

```
2930 REM
2940 FOR g=4 TO 11
2950 PRINT AT g,3;g$(g-3)
2960 NEXT g
2970 RETURN
2980 REM PRINT BANCO
2990 REM el que se esta trabajando
3000 GO SUB 3130
3010 PRINT AT 13,0;"Banco :"; INVERSE 1;banco; INVERSE 0;": ABCDEFGHIJKLMNOPQRST
U"
3020 PRINT AT 14,10;
3030 FOR i=144 TO 164
3040 PRINT CHR$ i;
3050 NEXT i
3060 GO SUB 3180
3070 GO SUB 3090
3080 RETURN
3090 REM COLOCA EL SET NORMAL
3100 POKE 23675,88
3110 POKE 23676,255
3120 RETURN
3130 REM COLOCA EL SET MARCADO
3140 REM POR DIRBAS
3150 POKE 23675,FN L(DIRBAS)
3160 POKE 23676,FN H(DIRBAS)
3170 RETURN
3180 REM SACA PIZARRA
3190 GO SUB 3130
3200 FOR I=1 TO 8
3210 PRINT AT 4+I-1,18;P$(I)
3220 NEXT I
3230 GO SUB 3090
3240 RETURN
3250 REM SACA LA PANTALLA DEL MENU
3260 PRINT AT 18,0;"1-Cambia banco.", "5-Edita pizarra.2-Set UDG.", "6-Extras."
3270 PRINT AT 20,0;"3-Copia.", "7-Borra.", "4-Save/Load.", "8-Basic."
3280 INPUT ""
3290 PRINT #1;" O,P,Q,A-Mover 1-8 Opciones"
3300 RETURN
3310 REM COPIA DEL UDG A LA PARRILLA
3320 GO SUB 3130
3330 DIM G$(8,8)
3340 FOR G=1 TO 8
3350 LET T=PEEK (USR C$+G-1)
3360 LET T2=128
3370 FOR I=1 TO 8
3380 LET g$(g,i)="A"
3390 LET T3=T-T2
3400 IF T3>=0 THEN LET T=T3: LET g$(g,I)="B"
3410 LET T2=T2/2
3420 NEXT I
3430 NEXT G
3440 GO SUB 3090
3450 GO SUB 2920
3460 RETURN
3470 REM ERROR EN UNA ENTRADA
3480 BEEP 1,0: PRINT #1;TAB 7; FLASH 1;"ERROR:OPCION ANULADA"
3490 PAUSE 200
3500 RETURN
3510 REM BORRADO DE LA PARTE DESTINADA AL MENU
3520 FOR I=16 TO 21
3530 PRINT AT I,0;"
3540 NEXT I
3550 INPUT ""
3560 RETURN
```

Antes de ver cómo podemos utilizar el programa, y qué opciones tiene, vamos a explicar su utilidad y para qué nos puede servir.

El programa DISEÑADOR DE CARACTERES PARA SPECTRUM está pensado como una herramienta que nos permitirá definir nuestros propios caracteres sin necesitar de un papel ni de un lápiz. Gracias a este programa podremos crear todos los caracteres que queramos y almacenarlos en cinta para su uso posterior. Es más, gracias a su sencillez, no tenemos por qué saber cómo se definen los caracteres de SPECTRUM, ya que nosotros sólo tenemos que encargarnos de dibujarlos en la pantalla.

Antes de introducir el programa hay que decir que las letras que aparecen subrayadas en el listado han de ser introducidas en modo GRAPHICS. Este modo de cursor se caracteriza porque nos permite incluir en el listado los caracteres definibles por el usuario (UDG) y los caracteres semigráficos. Para hacer que el cursor pase al modo GRAPHICS sólo hay que pulsar las teclas CAPS-SHIFT y 9 a la vez. Una vez hecho esto, el cursor se volverá una G mayúscula parpadeante. En ese momento podemos pulsar la tecla subrayada. En la pantalla nos puede aparecer cualquier cosa. Lo mismo podremos ver la tecla que hemos pulsado (esto pasa cuando dicho carácter no ha sido definido) que un marciano (en el caso de que lo hayamos definido anteriormente). Una vez tecleadas todas las letras subrayadas, para volver al modo normal del cursor sólo tenemos que volver a pulsar a la vez las teclas CAPS-SHIFT y 9.

Cuando ejecutamos el programa nos aparece un mensaje en la pantalla que nos dice que nos esperemos medio minuto. Una vez pasado ese tiempo (aproximadamente) nos aparecerá en la pantalla algo parecido a lo que podemos ver en la figura 1.

Lo que vemos en esta pantalla es lo siguiente:

— Arriba, a la izquierda, vemos una cuadrícula que está en blanco. Esta cuadrícula la utilizaremos para dibujar nuestro nuevo carácter. En la figura 1 se ha dibujado un cuadrado a modo de ejemplo. Esta cuadrícula es muy importante, pues todo el programa se centra en ella y en lo que dentro de ella ocurra.

— Arriba, a la derecha, nos encontramos con un cuadrado en blanco que tiene una etiqueta que dice PIZARRA. Eso es justamen-



te lo que es. En este cuadrado nosotros podremos dibujar lo que queramos utilizando los caracteres que hemos definido. Su utilidad estriba en que si queremos definir un gráfico que ocupe más de un carácter, como necesitamos saber cómo va quedando nuestro gráfico a tamaño real, en esta pizarra podremos dibujarlo para verlo.

— Debajo de estos dos cuadrados tenemos dos líneas, las cuales nos informan del banco que estamos utilizando y de los caracteres que hay en dicho banco. Un banco es una serie completa de caracteres que van desde la A hasta la U, todos ellos definibles. Como muchas veces necesitamos definir más de 21 caracteres, el programa dispone de 9 bancos distintos para crear hasta 200 caracteres. Las dos líneas que están junto al mensaje de información del banco que estamos utilizando nos informan de la forma que tiene cada uno de los caracteres que estamos definiendo. La primera fila nos dice la tecla que tendremos que pulsar para ver el carácter que se encuentra en la segunda fila. Al principio del programa, antes de haber definido ningún gráfico, las dos líneas coincidirán.

— Las cuatro líneas siguientes forman un menú y nos dicen las posibles opciones que tenemos y que podemos utilizar. Estas serán explicadas con más detalle un poco más adelante.

— La última línea de la pantalla nos dice las teclas que podemos utilizar para mover un pequeño cursor parpadeante que aparecerá dentro de la parrilla superior y que nos servirá para dibujar el gráfico. Estas teclas son:

- O - Izquierda
- P - Derecha
- Q - Arriba
- A - Abajo

ESPACIO - Fijar o borrar un punto según sea su estado.

TRUCOS Y RUTINAS BASICAS

Veamos ahora qué funciones podemos realizar con el programa aparte de dibujar nuestro gráfico en la parrilla. Para saberlo y para utilizarlas tenemos un menú de ocho opciones, cuyo significado es:

1. Nos permite cambiar el banco de caracteres con el que estamos trabajando.

2. Nos permite asignar la forma que acabamos de dibujar en la parrilla a un cierto carácter.

3. Copia la forma de un cierto carácter en la parrilla. Esta opción es buena para modificar un carácter que ya hemos definido.

4. Pulsando esta opción podemos grabar o leer desde cinta una serie de caracteres que hemos definido y que queremos leer o grabar.

5. Nos permite utilizar la pizarra para dibujar en ella con los caracteres que hemos definido.

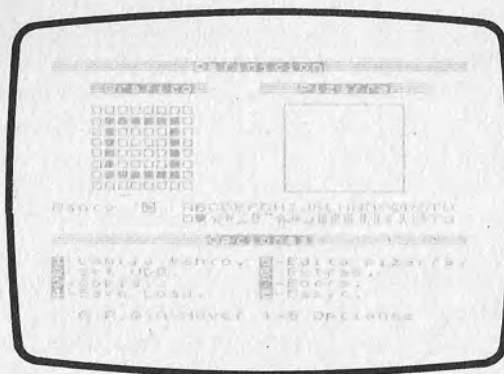
6. Pulsando esta opción nos aparecerá otro menú con más opciones. Estas las veremos un poco más adelante.

7. Borra la parrilla para volver a empezar. Lo que hubiésemos dibujado en ella se borraría y no quedaría almacenado.

8. Nos permite volver al BASIC.

Si hemos pulsado la opción 6, nos aparecerá otro menú en pantalla con otra serie distinta de opciones. Estas se pueden ver en la figura 2.

En este caso sólo tenemos seis opciones. Veamos para qué sirve cada una de ellas.



1. Nos permite volver al menú general. Este es del que hemos partido pulsando la opción seis para llegar a éste.

2. Nos invierte el contenido de la parrilla. Todos los puntos iluminados los apagará y todos los puntos apagados los iluminará.

3. Hace que el contenido de la parrilla cambie y se ponga como si fuese un espejo vertical del contenido anterior.

4. Realiza la misma función, pero en sentido horizontal.

5. Nos permite rotar el carácter, BIT a BIT, hacia la izquierda.

6. Nos permite rotar el carácter hacia la derecha.

Con estas instrucciones queda claro el funcionamiento del programa. Es conveniente que el lector lo utilice durante algunas horas para familiarizarse con él y para llegar a entender cómo funcionan y para qué sirven cada una de las opciones.

Proceso digital de señales



ON los últimos circuitos presentados deberemos estar ya al corriente de cómo adquirir las señales del mundo analógico externo, tanto de señales de audiofrecuencia como de cualquier tipo de señales cuya frecuencia no

sea muy elevada. El proceso que se ha indicado era de tipo básicamente analógico, mediante filtros y controlado por el ordenador como mero instrumento de selección y actuación.

El ordenador puede, sin embargo, emplearse como parte fundamental del proceso de las señales, pues una vez almacenada la información digitalizada puede tratarse como cualquier otro conjunto de datos. Vamos a presentar la forma de tratar las señales para conocer su contenido en las diferentes frecuencias y el proceso necesario para filtrar las que sea necesario eliminar o extraer para su utilización.

Análisis de señales periódicas

En casi todas las técnicas y en la física es muy frecuente encontrar fenómenos que presentan carácter periódico, es decir, fenómenos en los que los valores de las cantidades se repiten a intervalos regulares de tiempo. Estos fenómenos se representan por una fórmula matemática en la cual hay una variable que da el mismo valor para la función, para valores múltiplos de dicha variable.

Las funciones más conocidas que presentan esta propiedad son las funciones seno y coseno, cuyo período es 2π , es decir, se define como período el intervalo de la variable que produce la repetición del valor de la función.

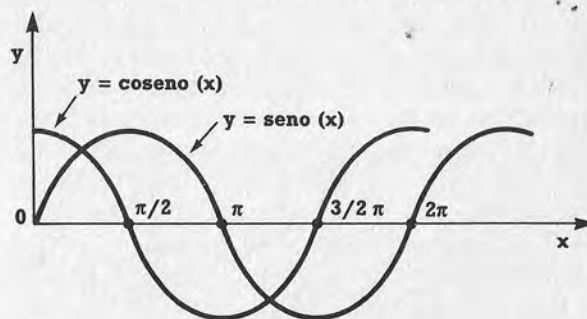


Fig. 1. Señales seno y coseno.

Supongamos que un determinado experimento relacionado con señales acústicas nos ha proporcionado una señal durante un determinado período de tiempo. Sería muy interesante poder conocer el contenido de los armónicos presentes en la señal para saber las características del instrumento que lo produjo. Con el analizador de señales de audiofrecuencia podríamos sacar una idea de la distribución de energía en cada una de las bandas en las que colocamos los filtros. Pero, ¿no podríamos utilizar el ordenador para hacer el análisis de manera más eficiente y con mayor resolución? La respuesta es afirmativa si programamos la máquina con el programa adecuado de análisis digital de señales.



Fig. 2. Señal física generada en un experimento.

El desarrollo en serie de armónicos de una señal periódica se ha realizado desde hace muchos años, mediante la fórmula de Fourier, que da los coeficientes de cada uno de los armónicos para producir la señal analizada. El análisis requiere la integración o suma del producto de la señal analizada por la senoide del armónico. El proceso consiste en multiplicar la señal por la senoide correspondiente en el intervalo analizado, y realizar la integración o suma del área encerrada por la curva en dicho intervalo.

Cuando la señal es obtenida experimentalmente, las operaciones de multiplicación y suma han de hacerse de forma aproximada utilizando herramientas que permitan representar las magnitudes en forma medible, como áreas o tensiones eléctricas. Durante muchos años se han utilizado los denominados planímetros, que mediante ruedas y mecanismos de precisión dan el valor del área encerrada en un recinto plano. En casos más elementales, el proceso consistía en contar cuadrados.

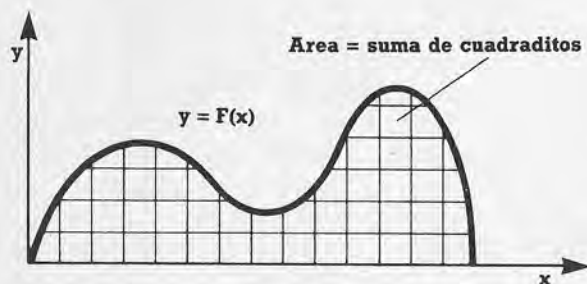


Fig. 3. Cálculo elemental del área bajo una curva.

Con la llegada de los ordenadores el proceso puede realizarse de forma totalmente automática y con la precisión que se desee. La transformación resultante se denomina Transformación Discreta de Fourier (DFT). Existe además un algoritmo que permite realizar la transformación de manera muy rápida, por realizarse las operaciones de forma iterativa utilizando operandos previamente calculados. Este algoritmo se denomina Transformada Rápida de Fourier (FFT=Fast Fourier Transform). Actualmente se emplea en todos los procesos de señal, en cualquiera de sus múltiples variantes. Existen incluso circuitos integrados de aplicación específica que realizan la transformación en tiempo real sobre los datos suministrados en sus entradas.

La Transformación de Fourier genera una secuencia de números que son los coeficientes de los armónicos de la señal. ¿Cuál es el interés de conocer estos valores en lugar de la señal original? El interés principal radica en que permite describir la señal de forma «resumida» con un conjunto muy limitado de números. En segundo lugar permite separar fácilmente las señales útiles del ruido que generalmente las acompaña, pues suele tener diferente contenido espectral. Además, este análisis es reversible, pues aplicando el algoritmo inverso podemos reconstruir la señal original, después de haber eliminado el ruido. Como última razón de interés es necesario citar que las operaciones de filtrado se reducen a meras multiplicaciones por la función de transferencia del filtro, para conocer la señal resultante.

El número de términos de la serie de Fourier que es necesario utilizar para representar una señal depende de las características propias de la señal. Un número reducido de componentes dará una buena representación si la señal es «poco brusca». Una señal con saltos contiene muchos armónicos y, por tanto, requiere una serie larga para su representación.

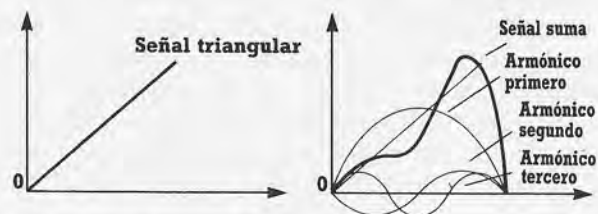


Fig. 4. Señal triangular y tres de sus armónicos.

Conviene hacer algunas aclaraciones adicionales antes de describir el programa que realiza la transformación. En el empleo de señales senoidales en los sistemas físicos se acostumbra a representar las correspondientes transformaciones originadas en los sistemas por las modificaciones en la amplitud y la fase de la señal de referencia. La señal a la entrada del sistema será generalmente una senoide de amplitud A y fase 0 . Al pasar por una parte del sistema presentará la amplitud B y la fase. Además, pueden aparecer distorsiones, por supuesto. Para referir al mismo origen de fase de la señal se acostumbra a representar mediante un diagrama vectorial la relación entre entrada y salida. Para indicar de otra manera esta diferencia de fases se

suele descomponer la señal de salida en dos componentes: una R que está en fase con la entrada y otra I que está separada 90° de ella. Una señal separada 90° de la función seno será una función coseno. A esta componente se la suele denominar «imaginaria», mientras que a la otra se la suele denominar «real». Por supuesto que las dos son completamente reales, pero en el sentido físico de generación de potencias solamente la componente que está en fase produce consumo.

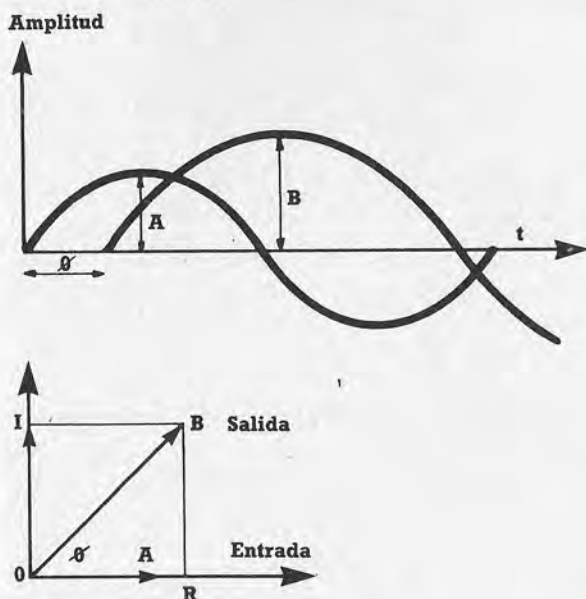


Fig. 5. Relación de señales entrada/salida.

Interesa tener estas consideraciones presentes para entender que el resultado de la transformación presenta una componente real y otra imaginaria, dentro del contexto indicado, pues las componentes resultantes pueden no estar en fase con la señal original y, por tanto, presentar una componente cosenoidal.

Programa de cálculo de FFT de una señal

Se muestra como ejemplo de uso de la FFT el proceso de una señal generada de forma sintética en las sentencias 10000 y siguientes. En un caso real deberíamos sustituir la subrutina indicada por una de lectura de señales del sistema físico y asignar a las componentes del vector X de componentes reales. El vector Y va asociado al X y contendrá a la sa-

lida la componente imaginaria de las transformadas.

También se muestra el proceso inverso, pues es idéntico al directo salvo en el signo del cálculo de ángulos y en el factor de escala. El listado corresponde a la programación para el IBM-PC y compatibles, pero puesto que no se usan instrucciones especiales de gráficos puede aplicarse a los demás ordenadores personales sin cambios. Los resultados se presentan de forma numérica y semigráfica, utilizando el posicionamiento de un * como indicación aproximada del valor. Para una primera idea es suficiente. Si se desea se pueden modificar las rutinas situadas de 5000 a 6060.

El número de puntos utilizado en la representación es 32, es decir, el correspondiente a muestrear una señal para poder recuperar hasta el armónico número 16. Se verá que al reconstruirla, como la señal original tenía armónicos superiores, aparecen ondulaciones correspondientes a los alias generados, además del pulso original de forma aproximada.

La rutina de FFT es general y puede emplearse con cualquier valor de P, teniendo en cuenta que el número de muestras ha de ser igual a 2^P , es decir, un número de potencia exacta de 2. Si el número obtenido en un experimento fuera inferior, se rellenará con ceros hasta completar el número necesario o se tomarán las muestras entre dos pasos por un mismo valor que aparezca como repetitivo, indicando la existencia de periodicidad.

Las líneas de la 1 a la 990 son el programa principal, donde puede apreciarse la forma de llamar a las subrutinas. De la 1000 a la 1600 está todo el cálculo de la transformada. Puede extraerse para otro programa, sólo requiere la definición de los vectores X e Y, el número de puntos y el ID para indicar si es directa o inversa. El cálculo empieza por una reordenación de los valores, para separar las componentes pares de las impares. El proceso principal consiste en la multiplicación de las señales muestrales por los valores de senos y cosenos de los ángulos del círculo dividido por el número de puntos de muestra. Se representa mediante un diagrama de «mariposa» en el que se aprecia que cada valor se multiplica por dos coeficientes para pasar a la fase siguiente.

EL TALLER DEL HARDWARE

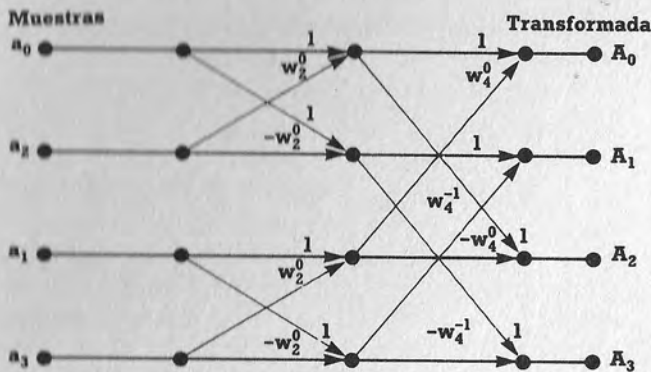


Fig. 6. Diagrama de cálculo de la FFT para 4 puntos.

Listado del programa ejemplo

```

10 REM      * * * * *
20 REM      * * *   EJEMPLO DE USO DE FFT   * * *
25 REM      * * * * *
30 TM=40:REM TAMAÑO MAXIMO DEL GRAFICO
40 P=5: REM CORRESPONDE A 32 MUESTRAS POR CICLO
50 B$=""
60 KEY OFF: CLS
70 N=2^P: PRINT " EJEMPLO DE TRATAMIENTO DIGITAL DE SEÑALES"
80 DIM X(N),Y(N),Z(N),INF(10),II(10)
90 DIM V(13),W(13)
100 GOSUB 10000:REM GENERA SEÑAL
105 FOR I=1 TO N: Z(I)=X(I): NEXT I
110 GOSUB 5000: REM CALCULA MAXIMO, MINIMO Y ESCALA
120 PRINT
130 PRINT "GRAFICO DE LA SEÑAL DE ENTRADA"
140 PRINT
150 GOSUB 6000: REM LISTADO GRAFICO
155 ID=1: REM TRANSFORMADA DIRECTA
160 GOSUB 1000: REM CALCULO DE FFT
170 FOR I=1 TO N
180   PRINT USING "X(##)=###.#####";I,X(I);
190   PRINT USING ",   Y(##)=###.#####";I,Y(I)
200 NEXT I
210 FOR I=1 TO N
220   Z(I)=SQR(X(I)*X(I)+Y(I)*Y(I))
230 NEXT I
240 GOSUB 5000: REM MAX, MIN Y ESCALA
250 PRINT
260 PRINT "GRAFICO DEL MODULO DE LA FFT"
270 PRINT
280 GOSUB 6000: REM LISTADO GRAFICO
290 REM CALCULA INVERSA, PARA VER RESULTADO
300 ID=-1
310 GOSUB 1000
320 FOR I=1 TO N
330   PRINT USING "X(##)=###.#####";I,X(I);
340   PRINT USING ",   Y(##)=###.#####";I,Y(I)
350 NEXT I
355 FOR I=1 TO N: Z(I)=X(I): NEXT I
360 GOSUB 5000: REM MAX, MIN Y ESCALA
370 PRINT
380 PRINT "GRAFICO DE LA SEÑAL RECONSTRUIDA"
390 PRINT
410 GOSUB 6000: REM LISTADO GRAFICO
990 END
999 REM SUBROUTINA FFT(ID,X,Y,P)
1000 PRINT: PRINT "CALCULO DE LA FFT": PRINT
1020 REM      REORDENA
1030 FOR I=1 TO P
1050   INF(I)=2^(P-I)
1060 NEXT I
1070 FOR L=1 TO N
1080   I=1
1090   FOR J=1 TO P
1100     IF (II(J)<2) THEN 1140
1110     II(J+1)=II(J)+1
1120     II(J)=0
1130   NEXT J
1140   FOR J=1 TO P

```

```

1150     I=I+II(J)*INF(J)
1160 NEXT J
1170 II(1)=II(1)+1
1180 IF L<I THEN 1250
1190 A=X(I)
1200 B=Y(I)
1210 X(I)=X(L)
1220 Y(I)=Y(L)
1230 X(L)=A
1240 Y(L)=B
1250 NEXT L
1260 REM                               INICIALIZACION
1270 PI=ID*3.141592653#
1280 M=1
1290 FOR I=1 TO P
1300     KM=N/(2*M)
1310     RS=SIN(PI/(2*M))
1320     R=-4*RS*RS
1330     VC=-.5*R
1340     VS=SIN(PI/M)
1350     FOR K=1 TO KM
1360         CD=VC
1370         SD=VS
1380         CS=1
1390         SN=0
1400         FOR L=1 TO M
1410             I1=L+(2*M*(K-1))
1420             I2=M+I1
1430             XT=X(I1)
1440             YT=Y(I1)
1450             XCS=CS*X(I2)
1460             XSN=SN*X(I2)
1470             YCS=CS*Y(I2)
1480             YSN=SN*Y(I2)
1490             X(I1)=XT+XCS+YSN
1500             Y(I1)=YT+YCS-XSN
1510             X(I2)=XT-XCS-YSN
1520             Y(I2)=YT-YCS+XSN
1530             CD=(R*CS)+CD
1540             CS=CS+CD
1550             SD=(R*SN)+SD
1560             SN=SN+SD
1570         NEXT L
1580     NEXT K
1590     M=2*M
1600 NEXT I
1610 RETURN
5000 REM                               CALCULA MAXIMO MINIMO Y ESCALA
5010 F=Z(1)
5020 G=Z(1)
5030 FOR I=2 TO N
5040     IF Z(I) > F THEN F=Z(I)
5050     IF Z(I) < G THEN G=Z(I)
5060 NEXT I
5070 ES=TM/(F-G)
5080 PRINT
5090 PRINT USING "MAX=####.##, MIN=####.##, ESCALA=###.##";F,G,ES
5100 PRINT
5110 RETURN
6000 REM                               SALIDA COMO GRAFICO
6010 FOR I=1 TO N
6020     H=1+INT((Z(I)-G)*ES)
6030     L$=LEFT$(B$,H)+"*"+LEFT$(B$,62-H)
6040     PRINT USING "###.#### ";Z(I);
6050     PRINT L$
6060 NEXT I
10000 REM                               GENERA LA SEÑAL
10010 X(1)=1: REM EJEMPLO DE PULSO TRIANGULAR
10020 X(2)=2
10030 X(3)=3
10040 X(4)=4
10050 X(5)=5
15000 RETURN

```

La relación entre la frecuencia de los armónicos y el período de muestreo es la siguiente: el armónico 0 corresponde al valor medio de la señal en el intervalo de muestreo

y los valores sucesivos corresponden a frecuencias múltiplos de la frecuencia de muestreo. En otras palabras X(0) dará el valor medio de la señal, X(1) e Y(1) las componentes

EL TALLER DEL HARDWARE

real e imaginaria del armónico primero, o fundamental correspondiente a la frecuencia $1/H$, siendo H el período completo de muestras. El tiempo entre muestras define el ancho de banda del espectro. La gráfica resultante resulta siempre simétrica, para datos reales de entrada, centrada sobre la componente $X(0)$. Por esta razón se suele representar la gráfica cen-

trada sobre el origen, presentando valores de frecuencia positivas y negativas. ¿Tiene sentido hablar de frecuencias negativas? Sí, el mismo que tiene hablar de tiempos negativos. Para un sistema que manipula datos almacenados podemos tomar como origen de tiempos cualquier punto y, por tanto, todo lo anterior será pasado y todo lo posterior futuro.

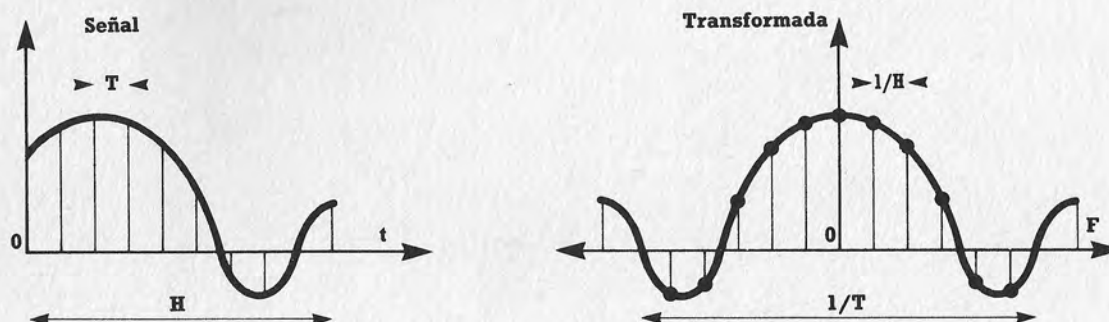


Fig. 7. Relación entre señal y transformación.

Ejemplo de ejecución del programa

Para mostrar los resultados que se obtienen al ejecutar el programa se ha listado el

resultado del análisis de una señal triangular. La transformada resultante se ha representado sólo en módulo. En el listado numérico aparecen las componentes real e imaginaria.

```
LOAD"FFT6
Ok
RUN
EJEMPLO DE TRATAMIENTO DIGITAL DE SEÑALES

MAX= 5.00, MIN= 0.00, ESCALA= 8.00

GRAFICO DE LA SEÑAL DE ENTRADA

1.0000 : *
2.0000 : *
3.0000 : *
4.0000 : *
5.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
```

```

0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *
0.0000 : *

```

CALCULO DE LA FFT

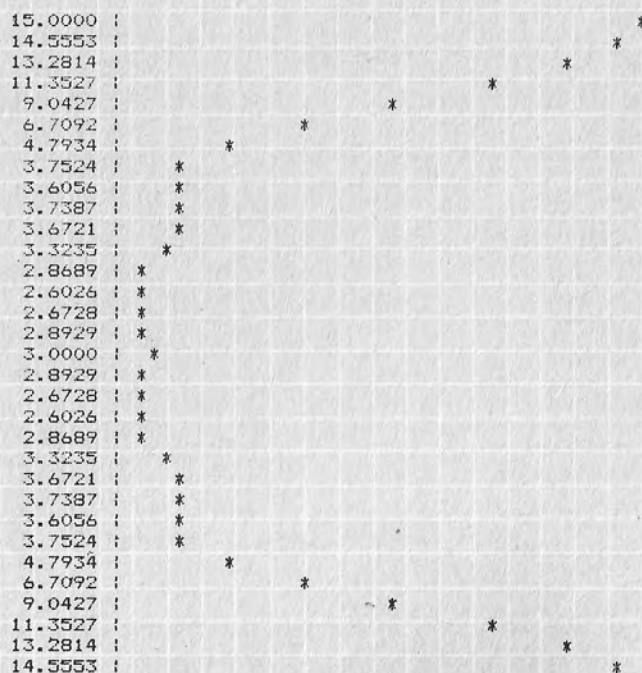
```

X( 1)= 15.00000,   Y( 1)=  0.00000
X( 2)= 12.59462,   Y( 2)= -7.29605
X( 3)=  6.49981,   Y( 3)=-11.58221
X( 4)= -0.50491,   Y( 4)=-11.34145
X( 5)= -5.41421,   Y( 5)= -7.24264
X( 6)= -6.49558,   Y( 6)= -1.67941
X( 7)= -4.05147,   Y( 7)=  2.56165
X( 8)= -0.06820,   Y( 8)=  3.75179
X( 9)=  3.00000,   Y( 9)=  2.00000
X(10)=  3.59600,   Y(10)= -1.02318
X(11)=  1.80883,   Y(11)= -3.19570
X(12)= -0.87158,   Y(12)= -3.20719
X(13)= -2.58579,   Y(13)= -1.24264
X(14)= -2.27006,   Y(14)=  1.27289
X(15)= -0.25717,   Y(15)=  2.66044
X(16)=  2.01973,   Y(16)=  2.07112
X(17)=  3.00000,   Y(17)=  0.00000
X(18)=  2.01972,   Y(18)= -2.07112
X(19)= -0.25717,   Y(19)= -2.66044
X(20)= -2.27006,   Y(20)= -1.27289
X(21)= -2.58579,   Y(21)=  1.24264
X(22)= -0.87158,   Y(22)=  3.20720
X(23)=  1.80883,   Y(23)=  3.19571
X(24)=  3.59600,   Y(24)=  1.02318
X(25)=  3.00000,   Y(25)= -2.00000
X(26)= -0.06821,   Y(26)= -3.75179
X(27)= -4.05147,   Y(27)= -2.56165
X(28)= -6.49559,   Y(28)=  1.67941
X(29)= -5.41421,   Y(29)=  7.24264
X(30)= -0.50491,   Y(30)= 11.34146
X(31)=  6.49981,   Y(31)= 11.58221
X(32)= 12.59462,   Y(32)=  7.29604

```

MAX= 15.00, MIN= 2.60, ESCALA= 3.23

GRAFICO DEL MODULO DE LA FFT



CALCULO DE LA FFT INVERSA

X(1)= 18.82469,	Y(1)= 0.00000
X(2)= 47.48519,	Y(2)= 6.46053
X(3)= 71.46106,	Y(3)= 8.04709
X(4)= 95.60875,	Y(4)= 2.66288
X(5)=124.90880,	Y(5)= -7.80591
X(6)=-30.32903,	Y(6)=-18.52761
X(7)=-19.74163,	Y(7)=-24.40106
X(8)= -7.97232,	Y(8)=-22.84002
X(9)= -0.08598,	Y(9)=-15.09953
X(10)= 1.16707,	Y(10)= -5.43346
X(11)= -3.30889,	Y(11)= 1.37685
X(12)= -9.72025,	Y(12)= 2.66712
X(13)=-13.73725,	Y(13)= -0.80626
X(14)=-13.01531,	Y(14)= -5.59827
X(15)= -8.41054,	Y(15)= -7.83187
X(16)= -3.22408,	Y(16)= -5.58875
X(17)= -0.99588,	Y(17)= -0.00000
X(18)= -3.22409,	Y(18)= 5.58876
X(19)= -8.41054,	Y(19)= 7.83187
X(20)=-13.01531,	Y(20)= 5.59826
X(21)=-13.73724,	Y(21)= 0.80627
X(22)= -9.72024,	Y(22)= -2.66713
X(23)= -3.30888,	Y(23)= -1.37685
X(24)= 1.16707,	Y(24)= 5.43345
X(25)= -0.08597,	Y(25)= 15.09953
X(26)= -7.97234,	Y(26)= 22.84002
X(27)=-19.74163,	Y(27)= 24.40106
X(28)=-30.32905,	Y(28)= 18.52760
X(29)=-35.09119,	Y(29)= 7.80590
X(30)=-32.39125,	Y(30)= -2.66285
X(31)=-24.53895,	Y(31)= -8.04708
X(32)=-16.51482,	Y(32)= -6.46054

MAX= 124.91, MIN= -35.09, ESCALA= 0.25

GRAFICO DE LA SEÑAL RECONSTRUIDA

18.8247 :	*
47.4852 :	*
71.4611 :	*
95.6088 :	*
124.9088 :	*
-30.3290 :	*
-19.7416 :	*
-7.9723 :	*
-0.0860 :	*
1.1671 :	*
-3.3089 :	*
-9.7202 :	*
-13.7372 :	*
-13.0153 :	*
-8.4105 :	*
-3.2241 :	*
-0.9959 :	*
-3.2241 :	*
-8.4105 :	*
-13.0153 :	*
-13.7372 :	*
-9.7202 :	*
-3.3089 :	*
1.1671 :	*
-0.0860 :	*
-7.9723 :	*
-19.7416 :	*
-30.3290 :	*
-35.0912 :	*
-32.3913 :	*
-24.5390 :	*
-16.5148 :	*

Fig. 8. Listado de la salida del ejemplo.

Los resultados pueden hacerse más atractivos utilizando la capacidad gráfica de los ordenadores. Veremos rutinas apropiadas

al tratar de utilizar el ordenador como osciloscopio digital.

■ Conclusiones

El proceso de análisis de una señal por sus componentes de Fourier implica que se considere periódica a la señal, con período igual al de la toma de datos o un divisor. En la práctica puede no resultar cierta la afirmación y presentarse efectos laterales que es necesario considerar. Si el cálculo de la FFT se efectúa sobre señales reales, para conocer su espectro es necesario «suavizar» el comienzo y el final de la toma de muestras, para evitar los efectos de borde, conocido como «efecto ventana». Se consigue el suavizado precisamente aplicando una función de ventana que tiene un máximo en el centro y decrece suavemente por los extremos.

Si la señal tratada es una parte de una señal más amplia en el tiempo, como puede ser la generada por un instrumento musical o la voz humana, pueden estimarse las compo-

nentes armónicas y su variación con el tiempo, para apreciar la evolución de los diferentes formantes.

Como experimentos a realizar con el programa presentado es conveniente empezar con señales senoidales de diferentes frecuencias, generadas por incrementos en la subrutina 10000, para apreciar el efecto de la superposición y su diferenciación mediante análisis. También es conveniente analizar señales generadas mediante digitalización con «lápiz y papel», a partir de ejemplos de señales reales de un libro. Impresionará en seguida la precisión obtenible con un humilde ordenador personal.

La utilización del muestro de señales y los programas de análisis adecuados permiten aumentar el rango de aplicaciones en que podemos emplear nuestra máquina y la potencia de los resultados.

NATURALEZA Y TECNOLOGIA

Efecto Doppler



CUANDO la fuente emisora de un movimiento ondulatorio, el observador del mismo, o ambos, están en movimiento relativo respecto al medio en el cual se propaga el citado movimiento, la frecuencia que del mismo percibe el observador es distinta a la emitida por la fuente. Este fenómeno se conoce con el nombre de efecto Doppler en honor a Christian Johann Doppler (1803-1853), quien lo puso de manifiesto por primera vez en 1842.

Supongamos que el observador y la fuente se mueven en la misma dirección con velocidades V_o y V_f . En el instante inicial están separados por una distancia d , y que la velocidad de propagación del movimiento ondulatorio es V .

El frente de onda emitido en el instante inicial en F , será recibido por el observador en O' , después de un cierto tiempo t , y se verifica que

$$FO' = d + OO' \text{ y que } V \cdot t = d + V_o \cdot t$$

La onda emitida en el instante $t=0$ alcanzará al observador en el instante

$$t = d / (V - V_o)$$

Transcurrido el tiempo correspondiente a un período T del movimiento ondulatorio, se emite otro frente de onda análogo al primero. En este tiempo, la fuente habrá recorrido el espacio

$$FF' = V_f \cdot T$$

y es en F' donde se hace la emisión del frente. El nuevo frente alcanzará al observador en el instante t' y en la posición O'' .

El espacio recorrido por el observador es

$$OO'' = V_o \cdot t'$$

y el recorrido por el segundo frente de onda es

$$F'O'' = FO + OO'' - FF' = d + V_o \cdot t' - V_f \cdot T$$

Este espacio, puesto que el tiempo que está viajando el segundo frente es $t' - T$, debe ser $V \cdot (t' - T)$

$$V \cdot (t' - T) = d + V_o \cdot t' - V_f \cdot T$$

con lo que el instante donde el segundo frente de onda alcanza al observador es

$$t' = (d + (V - V_f) \cdot T) / (V - V_o)$$



Fig. 1.

Por tanto, para el observador, el tiempo que transcurre entre dos frentes de onda análogos y consecutivos, o período de movimiento ondulatorio según su percepción es

$$T' = t' - T = (V - V_f) / (V - V_o) \cdot T$$

La relación entre la frecuencia que emite la fuente f y la percibida por el observador f' es

$$f' = (V - V_o) / (V - V_f) \cdot f$$

ya que la frecuencia es la inversa del período.

En general, las velocidades deben tomarse con signo positivo cuando el sentido de la velocidad es de llegada de las ondas al observador y negativo en caso contrario.

```

10 REM *****
20 REM * PROGRAMA : EFECTO DOPPLER *
30 REM * VERSION : 1.0 *
40 REM * PARA IBM-PC Y COMPATIBLES, *
50 REM * SPECTRUM, AMSTRAD, MSX Y *
60 REM * COMMODORE *
70 REM *****
100 PRINT"ESTE PROGRAMA CALCULA LA FRECUENCIA"
110 PRINT"PERCIBIDA POR EL OBSERVADOR DE UNA "
120 PRINT"FUENTE DE MOVIMIENTO ONDULATORIO EN"
130 PRINT"MOVIMIENTO ASI COMO EL PERIODO "
140 PRINT"PERCIBIDO."
150 PRINT"HAY QUE INTRODUCIR LOS SIGUIENTES "
160 PRINT"DATOS : VELOCIDAD DE PROPAGACION "
170 PRINT"DE LA ONDA, VELOCIDAD DE LA FUENTE"
180 PRINT"Y DEL OBSERVADOR, CONSIDERANDO "
190 PRINT"COMO POSITIVO EL MOVIMIENTO HACIA"
200 PRINT"LA DERECHA."
300 PRINT"DATOS"
310 PRINT"====="
320 INPUT"VELOCIDAD DE PROPAGACION";V
330 INPUT"VELOCIDAD DE LA FUENTE";F
340 INPUT"VELOCIDAD DEL OBSERVADOR";O
350 INPUT"FRECUENCIA DE LA SEÑAL";S
360 LET T=1/S
370 LET P=T*(V-F)/(V-O)
380 LET C=S*(V-O)/(V-F)
390 PRINT"LA FRECUENCIA PERCIBIDA ES DE ";C;" HERTZIOS"
400 PRINT"EL PERIODO PERCIBIDO ES DE ";P;" SEGUNDOS"

```

PROBLEMAS MATEMATICOS

■ Recursividad

Se dice que un objeto es recursivo cuando forma parte de sí mismo o se define en función de sí mismo. La recursión aparece no sólo en matemáticas, sino también en la vida diaria. Por ejemplo, es recursiva la imagen creada en una peluquería cuando nos colocamos entre dos espejos situados en planos paralelos.

La recursión es un medio particularmente poderoso en matemáticas, sobre todo en las definiciones.

La potencia de la recursión reside en la posibilidad de definir un número infinito de objetos mediante un enunciado finito. De igual modo, un número infinito de operaciones de cálculo puede describirse mediante un programa recursivo finito, incluso si este programa no contiene repeticiones explícitas. De cualquier forma, los algoritmos recursivos son apropiados principalmente cuando el proble-

ma a resolver, o la función a calcular, o la estructura de datos a procesar, están ya definidos de forma recursiva.

Los algoritmos recursivos son particularmente apropiados cuando el problema a resolver o los datos a tratar se definen de forma recursiva. Sin embargo, esto no significa que tales definiciones recursivas garanticen que la mejor forma de resolver el problema sea un algoritmo recursivo. De hecho, la explicación del concepto de algoritmo recursivo mediante ejemplos inapropiados ha sido la causa principal de que se haya creado una antipatía general hacia el uso de la recursión, asociándose recursión a ineficacia.

Como programas ejemplo para ilustrar el tema de la recursividad se han propuesto el cálculo del factorial de un número y la resolución del problema de Las Torres de Hanoi.

Se define como factorial de un número natural como el producto de ese número por todos los números naturales menores que él. Por definición, el factorial de 0 es 1. El programa consiste en un bucle (recursividad) donde se van obteniendo los productos sucesivos.

```

10 REM *****
20 REM * PROGRAMA : FACTORIAL *
30 REM * VERSION : 1.0 *
40 REM * PARA IBM-PC Y COMPATIBLES, MSX, *
50 REM * AMSTRAD, COMMODORE Y SPECTRUM *
60 REM *****

```

```

100 PRINT"ESTE PROGRAMA CALCULA EL FACTORIAL DEL NUMERO"
110 PRINT"NATURAL INTRODUCIDO"
120 INPUT"NUMERO";N
130 PRINT N;
140 FOR I=N-1 TO 1 STEP -1
145 IF I/15=INT(I/15) THEN PRINT:PRINT
150 LET N=N*I
160 PRINT "*" ;I;
170 NEXT I
175 PRINT:PRINT
180 PRINT="";N

```

La resolución del problema de Las Torres de Hanoi consiste en mover discos situados en un poste A a otro poste B utilizando como auxiliar un poste C, y teniendo en cuenta que no puede colocarse un disco encima de otro menor.

Por ejemplo, la salida del programa para el caso de cuatro discos sería

```

NUMERO DE DISCOS? 4
MOVER DISCO 1 DE A A C
MOVER DISCO 2 DE A A B
MOVER DISCO 1 DE C A B
MOVER DISCO 3 DE A A C
MOVER DISCO 1 DE B A A
MOVER DISCO 2 DE B A C
MOVER DISCO 1 DE A A C
MOVER DISCO 4 DE A A B
MOVER DISCO 1 DE C A B
MOVER DISCO 2 DE C A A
MOVER DISCO 1 DE B A A
MOVER DISCO 3 DE C A B
MOVER DISCO 1 DE A A C
MOVER DISCO 2 DE A A B
MOVER DISCO 1 DE C A B

```

Fig. 2.

```

10 REM *****
20 REM * PROGRAMA : TORRES DE HANOI *
30 REM * VERSION : 1.0 *
40 REM * PARA IBM-PC Y COMPATIBLES, MSX, *
50 REM * AMSTRAD, COMMODORE Y SPECTRUM *
60 REM *****
100 INPUT"NUMERO DE DISCOS";N
110 LET X$="A":LET Y$="B":LET Z$="C"
120 GOSUB 1000
130 END
1000 IF N=1 THEN PRINT"MOVER DISCO 1 DE ";X$;" A ";Y$ :LET N=N+1:RETURN
1010 LET N=N-1:LET D$=Y$:LET Y$=Z$:LET Z$=D$:GOSUB 1000
1015 LET D$=Y$:LET Y$=Z$:LET Z$=D$
1020 PRINT"MOVER DISCO ";N;" DE ";X$;" A ";Y$
1030 LET N=N-1
1040 LET D$=X$:LET X$=Z$:LET Z$=D$
1050 GOSUB 1000
1055 LET N=N+1
1057 LET D$=X$:LET X$=Z$:LET Z$=D$
1060 RETURN

```

SOCIEDAD

■ Historia de la música

La música es sin duda el arte que más audiencia tiene hoy en el mundo: la radio, la televisión y el cine tienen que servirse continuamente de ella y prácticamente no se pueden concebir sin su ayuda. Por otra parte, la música se convierte en un compañero inseparable del hombre, desde su infancia a su ve-

jez, y en todas las circunstancias de su vida. El estudio de la Historia de la Música es, desde esta perspectiva, algo absolutamente necesario para la formación del hombre.

La música como todo arte tiene una relación estrecha con el mundo que le rodea, hasta tal punto, que nunca se puede entender prescindiendo de su situación estética, social y artística de la sociedad en que surge esta música, y de la misma manera, este arte provendrá de las necesidades de expresión de cada cultura. Esto quiere decir que como la

APRENDER CON EL ORDENADOR

pintura, escultura, arquitectura o literatura, va a tener una función importante y diferente en cada época.

Por ello en el programa se propone un

test para que que compruebes el estado de tu cultura musical y en concreto, que compruebes si sabes encuadrar a los principales compositores de todas las épocas en su movimiento cultural.

```
10 REM *****
20 REM * PROGRAMA : HISTORIA DE LA MUSICA *
30 REM * VERSION : 1.0 *
40 REM * PARA IBM-PC Y COMPATIBLES, MSX, *
50 REM * AMSTRAD Y COMMODORE. *
60 REM *****
100 PRINT"EL OBJETO DE ESTE PROGRAMA ES ANALIZAR UN"
110 PRINT"ASPECTO DE SU CULTURA MUSICAL: SABER"
120 PRINT"ENCUADRAR MUSICOS EN SUS RESPECTIVOS"
130 PRINT"MOVIMIENTOS MUSICALES"
140 PRINT"PARA ELLO APARECERA EN PANTALLA EL NOMBRE"
150 PRINT"DE UN MUSICO Y TENDRA QUE RESPONDER CON "
160 PRINT"EL MOVIMIENTO MUSICAL AL QUE PERTENECE"
170 DIM M$(31,2)
175 LET R=0
180 FOR I=1 TO 31
190 FOR J=1 TO 2
200 READ M$(I,J)
210 NEXT J
220 NEXT I
230 FOR I=1 TO 10
240 LET N=INT(RND(1)*31)+1
250 PRINT"¿A QUE MOVIMIENTO MUSICAL PERTENECE ";M$(N,1);"?"
260 INPUT A$
270 IF A$=M$(N,2) THEN PRINT"BIEN":R=R+1:GOTO 290
280 PRINT"INCORRECTO. LA RESPUESTA CORRECTA ES ";M$(N,2)
290 NEXT I
300 PRINT"HA TENIDO ";R;" ACIERTOS"
1000 DATA "ALBINONI","BARROCO ITALIANO","SCARLATTI","BARROCO ITALIANO"
1010 DATA "VIVALDI","BARROCO ITALIANO","PACHELBEL","BARROCO ALEMAN"
1020 DATA "BACH","BARROCO ALEMAN","TELEMANN","BARROCO ALEMAN"
1030 DATA "PURCELL","BARROCO INGLES","HANDEL","BARROCO INGLES"
1040 DATA "RAMEAU","BARROCO FRANCES","GLUCK","BARROCO FRANCES"
1050 DATA "MARTIN SOLER","BARROCO ESPAÑOL","P. SOLER","BARROCO ESPAÑOL"
1060 DATA "HAYDN","CLASICISMO","MOZART","CLASICISMO"
1070 DATA "ARRIAGA","CLASICISMO","BEETHOVEN","SINFONISMO"
1080 DATA "SCHUBERT","ROMANTICISMO","MENDELSSOHN","ROMANTICISMO"
1090 DATA "SCHUMANN","ROMANTICISMO","BRAHMS","ROMANTICISMO"
1100 DATA "BRUCKNER","ROMANTICISMO","FRANCK","ROMANTICISMO"
1110 DATA "BERLIOZ","ROMANTICISMO","STRAUSS","ROMANTICISMO"
1120 DATA "CHOPIN","ROMANTICISMO","LISZT","ROMANTICISMO"
1130 DATA "WEBER","OPERA ROMANTICA","WAGNER","OPERA ROMANTICA"
1140 DATA "MEYERBEER","OPERA ROMANTICA","GOUNOD","OPERA ROMANTICA"
1150 DATA "OFFENBACH","OPERA ROMANTICA","BIZET","OPERA ROMANTICA"
1160 DATA "ROSSINI","OPERA ITALIANA","BELLINI","OPERA ITALIANA"
1170 DATA "DONIZETTI","OPERA ITALIANA","VERDI","OPERA ITALIANA"
1180 DATA "PUCCINI","OPERA ITALIANA","BRETON","LIRICA ESPAÑOLA"
1190 DATA "CHAPI","LIRICA ESPAÑOLA","CHUECA","LIRICA ESPAÑOLA"
1200 DATA "VIVES","LIRICA ESPAÑOLA","SOROZABAL","LIRICA ESPAÑOLA"
1210 DATA "MORENO TORROBA","LIRICA ESPAÑOLA","GLINKA","NACIONALISMO RUSSO"
1220 DATA "BORODIN","NACIONALISMO RUSSO","RIMSKY-KORSAKOV","NACIONALISMO RUSSO"
1230 DATA "MUSSORGSKI","NACIONALISMO RUSSO","TCHAIKOVSKI","NACIONALISMO RUSSO"
1240 DATA "SIBELIUS","NACIONALISMO ESCANDINAVO",
      "GRIEG","NACIONALISMO ESCANDINAVO"
1250 DATA "KODALY","NACIONALISMO HUNGARO","BARTOK","NACIONALISMO HUNGARO"
1260 DATA "DEBUSSY","IMPRESIONISMO","RAVEL","IMPRESIONISMO"
1270 DATA "SARASATE","NACIONALISMO ESPAÑOL","ALBENIZ","NACIONALISMO ESPAÑOL"
1280 DATA "GRANADOS","NACIONALISMO ESPAÑOL","FALLA","NACIONALISMO ESPAÑOL"
1290 DATA "TURINA","NACIONALISMO ESPAÑOL","MAHLER","ROMANTICISMO"
1300 DATA "STRAVINSKI","EXPRESIONISMO","SCHOENBERG","DODECAFONISMO"
```

Modificaciones para otros equipos

SPECTRUM

LINEA 170 DIM M\$(31,2,25)

PARA LOS MAS JOVENES

■ Animales vertebrados

En esta ocasión hemos querido dar un repaso a tus conocimientos sobre la fauna ver-

tebrada. Antes de nada vamos a repasar ciertos conceptos.

Los animales vertebrados tienen huesos que forman su esqueleto. Se distinguen bien los huesos de la cabeza y los que forman la columna vertebral. Esta constituye una fila de huesos especiales llamados vértebras, que van a lo largo del cuerpo.

Los MAMÍFEROS tienen su cuerpo cubierto de pelos total o parcialmente, las hembras tienen mamas para alimentar a sus crías, tienen respiración pulmonar, su temperatura es constante y pueden ser PLACENTARIOS y MARSUPIALES.

Las AVES tienen el cuerpo cubierto de plumas, las extremidades anteriores están transformadas en alas, se apoyan sobre dos patas, tienen boca con pico, y además buche y molleja. Respiran por pulmones y nacen de huevos. Atendiendo al pico y a las alas se clasifican en PALMÍPEDAS, ZANCUDAS, PAJAROS, RAPACES, CORREDORAS y PRENSORAS.

Los REPTILES son vertebrados terrestres, tienen extremidades cortas o carecen de ellas, su piel está cubierta de escamas, escudos o placas óseas, se aletargan en invierno, respiran por pulmones, nacen de huevos y se clasifican en OFIDIOS, SAURIOS y QUELONIOS.

Los ANFIBIOS tienen la piel desnuda, húmeda y escurridiza, poseen cuatro patas, se aletargan en invierno, respiran primero por branquias y luego por pulmones y por la piel, nacen de huevos y sufren metamorfosis y se clasifican en ANUROS y URODELOS.

Los PECES son vertebrados acuáticos, tienen aletas pares e impares para nadar, su cuerpo está cubierto de escamas, respiran por branquias nacen de huevos y hay dos grupos importantes ESQUELETO OSEO y ESQUELETO CARTILAGINOSO.

El programa te preguntará a qué tipo y a qué clase pertenecen diez animales elegidos al azar.

```

10 REM *****
20 REM * PROGRAMA : ANIMALES VERTEBRADOS *
30 REM * VERSION : 1.0 *
40 REM * PARA IBM-PC Y COMPATIBLES, MSX, *
50 REM * COMMODORE Y AMSTRAD *
60 REM *****
100 PRINT"CON ESTE PROGRAMA PODRAS VERIFICAR"
110 PRINT"TUS CONOCIMIENTOS DE LOS ANIMALES "
120 PRINT"VERTEBRADOS. PARA ELLO, EL ORDENADOR"
130 PRINT"TE PREGUNTARA QUE TIPO DE ANIMAL ES"
140 PRINT"EL QUE APARECE EN PANTALLA Y A QUE "
150 PRINT"CLASE PERTENECE. ASI, POR EJEMPLO,"
160 PRINT"SI APARECE EN PANTALLA 'GATO' A LA"
170 PRINT"PREGUNTA DE TIPO TENDRAS QUE RESPONDER"
180 PRINT"MAMIFERO Y A LA PREGUNTA DE CLASE "
190 PRINT"RESPONDERAS PLACENTARIOS."
200 PRINT
210 DIM T$(5),C$(15),A$(30),A$(30,2)
300 FOR I=1 TO 5
310 READ T$(I)
320 NEXT I
330 FOR I=1 TO 15
340 READ C$(I)
350 NEXT I
360 FOR I=1 TO 30
370 READ A$(I)
380 READ A$(I,1)
390 READ A$(I,2)
400 NEXT I
480 LET R=0
490 FOR I=1 TO 10
500 N=INT(RND(1)*30)+1
510 PRINT"¿DE QUE TIPO ES EL SIGUIENTE ANIMAL? ";A$(N)
520 INPUT R$
530 IF R$=T$(A(N,1)) THEN PRINT"BIEN":R=R+1:GOTO 550
540 PRINT"RESPUESTA INCORRECTA"
550 PRINT"¿A QUE CLASE DE ";T$(A(N,1));" PERTENECE?"
560 INPUT R$
570 IF R$=C$(A(N,2)) THEN PRINT"BIEN":R=R+1:GOTO 590
580 PRINT"RESPUESTA INCORRECTA"
590 PRINT"LA RESPUESTA COMPLETA ES ";A$(N);" ";T$(A(N,1));" ";C$(A(N,2))
600 NEXT I
610 PRINT"HAS TENIDO ";R/2;" ACIERTOS"
900 DATA MAMIFERO,AVE,REPTIL,ANFIBIO,PEZ
910 DATA PLACENTARIO,MARSUPIAL,PALMIPEDA,ZANCUDA,PAJARO

```

```
920 DATA RAPAZ,CORREDORA,PRESORA,OFIDIO,SAURIO,QUELONIO
930 DATA ANURO,URODELO,QSEO,CARTILAGINOSO
1000 DATA PERRO,1,1,GATO,1,1,VACA,1,1,CANGURO,1,2
1010 DATA PATO,2,3,GAVIDOTA,2,3,GRULLA,2,4,GARZA,2,4
1020 DATA GORRION,2,5,CANARIO,2,5,BUITRE,2,6,AGUILA,2,6
1030 DATA AVESTRUZ,2,7,CASUARIO,2,7,LORO,2,8,PERIQUITO,2,8
1040 DATA CULEBRA,3,9,BOA,3,9,LAGARTO,3,10,COCODRILO,3,10
1050 DATA TORTUGA,3,11,GALAPAGO,3,11,RANA,4,12,SAPO,4,12
1060 DATA SALAMANDRA,4,13,TRITON,4,13,MERLUZA,5,14
1070 DATA SARDINA,5,14,TIBURON,5,15,RAYA,5,15
```

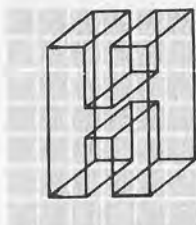
Modificaciones para otros equipos

SPECTRUM

LINEA 210 DIM
T\$(5,15),C\$(15,15),A\$(30,15),A(30,2)

PEQUEÑA HISTORIA DE LA INFORMATICA

■ El nacimiento de algunas de las más conocidas compañías de ordenadores



EMOS visto que durante siglos apenas si hubo avances en automatizar los cálculos. Sin embargo, en cincuenta años tuvo lugar una verdadera revolución y se crearon mercados casi imprevisibles unos años antes. Las

compañías que supieron ver que estaba naciendo un nuevo mercado para las máquinas calculadoras y tabuladoras son, casi sin excepción alguna, las que se han convertido en principales fabricantes del nuevo producto, el ordenador. El paso de un tipo de máquinas a otro no fue tampoco fácil, ni sin escollos. Ni tampoco los ordenadores tienen mucho que ver con esas ruidosas máquinas de tarjetas perforadas. Pero han sido un paso fundamental sin el cual, posiblemente, no se hubiera llegado a los ordenadores de nuestros días. No debemos olvidar que la idea de almacenar información en las tarjetas fue providencial, en su momento, y ha tenido vigencia hasta hace muy pocos años; es más, la sigue teniendo todavía, en cierto modo, ya que en nuestros días se sigue almacenando la información en medios externos (eso sí, algo más modernos en su concepción) como cintas, discos, etc.

Comencemos por un personaje conocido: Hollerith. El hombre que utilizó y popularizó las tarjetas perforadas, dejó de trabajar para el gobierno en 1896. Ya explicamos que el gobierno intentó sustituirlo por otro personaje, que también llegó a importantes cotas personales, pero seguro que en aquellos mo-

mentos les resultaba bastante más barato, y además utilizaba máquinas del propio gobierno, mientras que Hollerith les alquilaba las suyas propias, y además a precios cada vez más altos.

Hollerith fundó su propia compañía, la Tabulating Machines Corporation, que fue una compañía exitosa. Sus productos se vendieron por todo el mundo, y adquirió prestigio. Con el tiempo, la compañía fue creciendo, y Hollerith decidió unir su compañía con otras compañías también importantes: la Computing Scale Company of America, que fabricaba balanzas automáticas, y la Time Recording Company (fabricante de relojes de tipo industrial y para edificios, etc.). El nombre elegido para la fusión de las tres compañías fue CTR (Computing Tabulating Recording Company). En esta compañía trabajó Thomas J. Watson, que más tarde llegaría a presidente, en 1914, cuando la compañía tenía casi mil quinientos empleados.

A finales del siglo pasado, los acontecimientos ocurrían con mucha más rapidez que en siglos anteriores. Coches de caballos se entremezclaban con modernos automóviles por las estrechas calles mal pavimentadas en la mayor parte de los casos. Las empresas se modernizaban poco a poco, pero este halo de tiempos nuevos no era compartido por todos. Muchos presidentes o directores de empresa se aferraban a métodos antiguos, y algunos acababan con sus empresas, prósperas en otros tiempos, pero que eran «adelantadas» por otras empresas, en un principio menos sólidas.

Thomas Watson fue en sus primeros años al frente de la nueva empresa un hombre innovador y moderno. (No ocurrió lo mismo años después, pero esto lo veremos más adelante.) Cuando tuvo que apostar por las

PEQUEÑA HISTORIA DE LA INFORMATICA

tarjetas perforadas, no lo dudó. Les auguró un futuro brillante, aunque no muchos más compartieron su opinión. En 1917 se decidió a cambiar el nombre de una de las ramas de la compañía (la rama canadiense) y la llamó International Business Machines (IBM). Para el año 1924 la compañía era ya enorme, contando con más de 300.000 empleados en varios países. Dado el éxito del camino comercial emprendido, Watson, seguro de haber acertado, dio un paso más cambiando el nombre de la compañía y asignándole el de la rama canadiense. IBM estaba caminando con firmeza en el mundo de los negocios, a nivel multinacional.

Volvamos por un momento al pleito entre el gobierno de los Estados Unidos y Hollerith. El sustituto que puso el gobierno al frente de sus máquinas tampoco era un bonachón sin aspiraciones económicas. James Powers se ocupó del Censo de 1910, pero en 1911 ya tenía su propia compañía, la Accounting and Tabulating Machines Company, que fabricaba máquinas tabuladoras, también de tarjetas perforadas. Esta compañía pasó por muchas vicisitudes y fusiones, de las que reseñaremos la fusión con la Compañía de Máquinas de Escribir Remington (Remington Typewriter Company) cuyo presidente era entonces James Rand. Y así, en 1927, la compañía pasó a llamarse Remington Rand Corporation. De otra fusión posterior con la Sperry Gyroscope Company, ya (mucho más tarde, en 1950), surgió la Sperry Rand.

Acabamos de ver que las dos compañías fabricantes de ordenadores más importantes a nivel mundial salieron de un mismo punto, el Censo de los Estados Unidos.

Pero veamos qué ocurría en esas fechas en Francia y en Inglaterra, dos de las naciones europeas más industrializadas.

Uno de los pioneros en la fabricación de máquinas tabuladoras que utilizaban tarjetas perforadas fue el ingeniero noruego Frederik Bull. Bull, junto con otro ingeniero, Knut Kruesen, crearon una compañía en 1922, pensando en utilizar unas patentes de Bull anteriores a 1914. Sus máquinas se vendieron sobre todo en los países nórdicos, pero algunas también se comercializaron en Suiza. A la muerte de Bull, en 1925, un grupo suizo, Egli, compró las patentes y en 1929 creó la compañía Egli-Bull. Esta compañía no duró mucho tiempo, y en 1932 fue absorbida por otro grupo financiero, esta vez francés, la Caillies, adoptando un nuevo nombre, basado en el creador de la mayor parte de las patentes: Bull. La presidencia de

la nueva compañía, Societé des Machines Bull, fue para un personaje muy conocido en los ambientes militares e industriales de la época, el coronel Rimalho, que era gran aficionado a las armas, e incluso había diseñado él mismo una pistola que tuvo bastante éxito durante la Primera Guerra Mundial. El coronel Rimalho, además de experto en armas, también era un organizador nato, y llevó su compañía con habilidad. La empresa que competía con la Bull era en aquel momento casi únicamente IBM, ya que las máquinas de Bull eran bastante fiables y además muy rápidas, casi tanto como las de IBM, las más rápidas de aquel momento. La compañía Bull tuvo serias dificultades durante la Segunda Guerra Mundial, pero supo sobrevivir, y se recuperó.

Mientras tanto, en Gran Bretaña no estaba claro que el mercado empresarial pudiera absorber tantas máquinas innovadoras como al otro lado del océano (naturalmente considerando que la población es mucho menor). En 1915 fue creada la Powers Samas Company, dirigida principalmente para vender su producción al gobierno. Una rama de esta compañía, la Powers Americana, que tenía ciertos convenios con la Remington, siguió manteniéndolos con la compañía posterior, la Remington Rand. Sin embargo, los ingleses, al menos a nivel empresarial, no estaban ni mucho menos tan interesados como los norteamericanos en invertir dinero para eliminar tareas burocráticas. Sin embargo, la primera máquina popular a nivel empresarial fue una tabuladora creada por la Powers Samas, siendo su presidente Charles Foster.

En 1920 la CTR estaba subiendo peldaños a una velocidad vertiginosa. Y fue el momento adecuado para crear una nueva subsidiaria al otro lado del mar. La nueva compañía se llamó BTM (British Tabulating Machines). No conocemos las razones, pero después de muchos años (veintisiete de su creación), la BTM se independizó de la casa madre. Bastantes años después (1959) las dos compañías británicas más importantes, Powers Samas y BTM, sufrieron otra nueva fusión para crear la International Computers and Tabulators.

Pero ¿qué ocurría en los dos primeros años del siglo en España? ¿Nadie se interesaba por la automatización? A principios de siglo, España era un país pobre, poco industrializado, con una riqueza mal repartida y muchos problemas internos. No es que los demás países no tuvieran problemas de la misma índole, aunque de distinta importancia, pero

comparativamente estábamos más lejos de lo que hoy estamos de los países europeos industrializados. Es, pues, natural que pocos se interesasen por la automatización, existiendo problemas serios de industrialización.

Sin embargo, el ingenio español nunca muere. Y con los pocos medios de que han dispuesto generacionalmente los españoles para la investigación, aquellos que han tenido talento lo han aprovechado, bien en España, con mil dificultades, bien en el extranjero, con algo más de ayuda.

Leonardo Torres Quevedo fue uno de estos españoles insignes. Además fue probablemente el primero que utilizara la palabra

¿Sabía usted que...

¿Sabía usted que la primera máquina de jugar al ajedrez apareció en Madrid en 1911?

Efectivamente, y su inventor era también español, Leonardo Torres Quevedo. Daremos en estas cortas líneas una pequeña nota sobre su vida de ingeniero. Nació en un pequeño pueblo de Santander (Santa Cruz de Iguna), en 1852; murió en Madrid, en 1936. Fue muy considerado en ambientes universitarios de fuera de España, recibiendo el título de Doctor Honoris Causa por las Universidades de París y Coimbra. Como ingeniero, inventó el telekino, sistema para dirigir barcos a distancia, que utiliza ondas hertzianas. El telekino tuvo mucho éxito; el propio Torres Quevedo hizo las pruebas en El Abra de Bilbao, y la autora de estas líneas ha oído muchas veces a su abuelo contar la experiencia de cuando el señor Torres Quevedo realizó unas pruebas, moviendo una embarcación vacía por toda la ría, ante el estupor de los asistentes.

autómata, o automatización. Torres estaba muy interesado en la automatización. Muy considerado a nivel mundial, dirigió el Centro de Ensayos de Aeronáutica, avanzadilla de la técnica española, pasando más tarde a dirigir el Laboratorio de Automática. En uno de sus artículos, Torres Quevedo indica que los « autómatas deben ser capaces de discriminar, es decir, ser capaces de tener en cuenta las impresiones recibidas hasta ese momento, para pasar a realizar a continuación la acción requerida. Deben ser capaces de imitar el comportamiento humano, regulando sus acciones a las

impresiones recibidas, y adaptando su comportamiento a las circunstancias del momento». En el mismo artículo, Torres Quevedo se refiere a Descartes, cuando explicaba que un autómata no puede hablar «razonablemente». Nuestro insigne ingeniero dice: «Descartes consideraba la idea de que el autómata pudiera llegar a razonar, cuando es el constructor el que razona por él.»

En 1938, en Alemania, Konrad Zuse se ocupaba de construir un calculador, con lógica binaria. Este sistema estaba comenzando a estudiarse para ser utilizado en calculadoras, y ya tenía algunos adeptos, sobre todo en Francia, donde Louis Couffignal defendía su tesis de la utilización de sistema binario para la creación de calculadoras programables. No era el único. El ingeniero español Torres Quevedo se había interesado también, y un geómetra de prestigio, Maurice D'Ocagne, escribió un libro sobre las nuevas máquinas de calcular. Pero volvamos a Zuse. Su primer calculador, el Z1, aunque utilizaba un sistema binario, era absolutamente mecánico. Realmente su máquina no llegó a funcionar nunca. El segundo calculador, creado algo después, el Z2, también era mecánico. Y su resultado no fue tampoco bueno. El Z3, algo posterior, ya utilizaba relés, y fue construido en 1941. Este embrión de ordenador, y su versión posterior, el Z4 dieron un resultado excelente. Sin embargo, era un época difícil. La Segunda Guerra Europea congeló el trasvase de información entre los países, y supuso un enorme despilfarro de esfuerzo, ya que muchos descubrimientos fueron realizados por varias personas casi al mismo tiempo, en distintos países. (Aunque el avance de la técnica en tiempo de guerra sea siempre espectacular.)

Zuse recibió ayuda del Instituto Alemán de Investigación Aeronáutica, para la creación de su Z4. Esta industria lo utilizó ampliamente en sus aviones, junto con otros productos e instrumentos, también ideados por Zuse. El Z4 fue destruido en una incursión aérea, pero en su corta vida fue utilizado ampliamente para numerosos cálculos de todo tipo, desde controles de producción hasta cálculos para una bomba que era liberada desde un avión, y desde ese momento guiada por control remoto.

Zuse fue un autodidacta. No había leído a Babbage y apenas a Descartes, aunque, desde luego, sí había estudiado a Leibniz a fondo. (Es importante hacer notar que Zuse era un nazi convencido, que se ocultó en 1945 para salir a la luz pública en 1948, ya pasados los

PEQUEÑA HISTORIA DE LA INFORMATICA

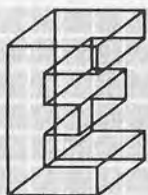
primeros rencores de la guerra.) Sus ideas políticas no le impidieron ser un cerebro brillante. Introdujo en su ordenador la «coma flotante» e inventó un lenguaje de programación universal algorítmico, pasando a programar sus máquinas con pericia. Pero la guerra lo dejó obsoleto, ya que sus soluciones no fueron conocidas en el momento por los otros países avanzados en estas técnicas, y cuando pudo rehabilitarse y volver a su trabajo (1950) los avances tecnológicos se habían basado en soluciones muy diferentes de las suyas.

En 1950 Zuse montó una pequeña compañía, que, tiempo después, fue absorbida por

la potente Siemens AG, fabricante de instrumentos eléctricos y electrónicos, ordenadores, y muchas otras cosas más.

Terminaremos esta pequeña disertación con una polémica: ¿cuál fue el primer ordenador, el Z4 de Konrad Zuse o el EINAC norteamericano? Nosotros no sabemos contestar. Los dos utilizaban técnicas parecidas (relés, etcétera), pero eran dos productos muy diferentes: la Z4, una máquina relativamente pequeña, y el ENIAC, enorme. ¿Pueden considerarse ambos dentro del mismo esquema? Opte el lector por la opción que prefiera.

■ El videotexto



L videotexto tiene como objetivo fundamental transformar el televisor doméstico en terminal de ordenador. De esta forma, el gran público podrá tener acceso a servicios informáticos amplísimos, que podrían resolverle

desde pequeños problemas hasta consultas más sofisticadas, como, por ejemplo, recabar cierto tipo de información de difícil acceso, y cuya obtención, en ciertos casos, lleva un tiempo considerable. También pueden tener fines didácticos.

La idea primitiva fue la que acabamos de exponer. Poner en contacto al gran público con las grandes redes de bases de datos existentes. Y era una idea ambiciosa. La información podía obtenerse por dos sistemas, un sistema interactivo, o simplemente difundido. Los métodos pensados fueron el teléfono y la difusión por televisión, al que llamaremos desde ahora teletexto.

Naturalmente, la difusión de información por televisión requería sistemas nuevos, agradables de ver y cómodos. Y es el momento en el que se acrecienta el interés por los gráficos y otros tipos de presentación en color. Los primeros gráficos y dibujos que se creaban utilizaban simplemente caracteres especiales (mosaicos gráficos) para la presentación de la información. Estos caracteres especiales es muy frecuente que formen parte de alguna ampliación del conjunto de caracteres estándar con que contaba el ordenador de que se trataba. Cada uno de estos caracteres mosaico está formado por una de las seis posibilidades que se pueden obtener dividiendo un rectángulo en seis zonas iguales,

pudiendo tener a su vez cada zona un estado diferente: o blanco o negro. Cuando se trabaja con color, se pueden considerar dos colores distintos, uno «de texto» y otro «de fondo», pero el problema es sustancialmente el mismo.

Como las necesidades del momento tampoco requerían mayores avances técnicos, y además el mercado no estaba maduro para recibirlo, el trazado de diagramas y otros dibujos con estas fuentes adicionales de caracteres constituía una técnica aceptable.

Sin embargo, en nuestros días este sistema es ya burdo (ya que existen muchos otros infinitamente mejores), y aunque sigue utilizándose por estar disponible fácilmente en algunos ordenadores, va sustituyéndose paulatinamente por otros sistemas. Citaremos tres cuyas ventajas e inconvenientes radican principalmente en la utilización que vaya a hacerse de ellos: el sistema de mosaico, el sistema geométrico, y, finalmente, el sistema fotográfico.

El primer sistema es casi exactamente aquel del que hemos estado hablando unas líneas más arriba. Sin embargo, ha sufrido grandes mejoras, pudiendo disponerse de varios conjuntos de caracteres adicionales. A su vez, cada carácter puede estar formado de forma distinta a la antes descrita, como, por ejemplo, mediante zona en forma de diagonal, etc., con lo que se pueden obtener caracteres muy diferentes, y, lo más importante, creando caracteres con posibilidades de ser redefinidos dinámicamente (también llamados caracteres DRCS —Dynamically Redefinable Character Sets—). Este último sistema permite definir los caracteres de forma absolutamente arbitraria, e incluirlos en otro conjunto de caracteres fuente adicional. Así pues, son caracteres absolutamente «a la medida» de cada aplicación

TEMAS MONOGRAFICOS DE VANGUARDIA

especial, disponiendo además de la ventaja de poder ser almacenados en la memoria del ordenador, de donde serán obtenidos llamando a la fuente de caracteres tantas veces como sea necesaria. (Piénsese, por ejemplo, en aplicaciones matemáticas que utilizan símbolos muy especiales, en química, en música, etc.)

Otro posible enfoque del problema es utilizar técnicas geométricas basadas en ciertas posibilidades de que disponen algunos terminales gráficos que trabajan en alta resolución (aunque no es imprescindible que sea alta resolución). El sistema se basa en considerar la información compuesta por multitud de atributos, que son codificados para su utilización. Entre estos atributos están líneas, arcos, círculos, polígonos, etc., por un lado, y por otro, los atributos del color. Este sistema de presentar las imágenes (al menos en principio), es independiente de la resolución en la que se está trabajando, y además presenta la ventaja de que se puede manipular utilizando sistemas gráficos y de animación. Por ello, es un sistema bastante eficaz. Sus detractores (entre los que se encuentran aquellos que utilizan o fabrican sistemas de las otras opciones posibles) argüirían que el decodificador que tratara la información de las imágenes debería ser muy sofisticado, y que, para conseguir crear imágenes con realismo, hace falta ser un verdadero artista.

La tercera opción es, desde luego, la más realista. Se utiliza en aquellos casos en los que las opciones anteriores no han ofrecido la resolución adecuada, o cuando sea necesario que las imágenes tengan un realismo más acusado. En este caso, la imagen se codifica línea por línea y punto por punto, para ser enviada o almacenada. Así, se obtiene una imagen de «níveles de gris» (ya explicamos este punto al tratar el tema de la visión artificial) o con una información de los colores. El principal problema que presenta esta opción es que la información correspondiente a cada imagen está compuesta por muchos datos que deben ser procesados, de forma que la velocidad utilizada por el videotexto supone una limitación enorme. Se han intentado solventar los problemas, limitando el tamaño de las imágenes y también mediante un sistema de codificación moderno (transmisión progresiva). Es evidente que con este sistema se pueden obtener imágenes más reales y perfectas que con los sistemas anteriores, pero las complicaciones que lleva implícitas hacen que sólo sea preferido en las ocasiones en las que los sistemas

de mosaico o geométrico no cumplan las especificaciones necesarias.

El videotexto en estos momentos está pasando por una crisis de identidad. Existen muchas tecnologías y sistemas que compiten y coexisten con él, siendo las diferencias poco claras. Así, existen sistemas educativos informatizados (Plato), otros sistemas de juegos electrónicos que utilizan el televisor doméstico (Playcable), bancos de datos públicos o privados, redes de ordenadores personales, etc. ¿El futuro previsible del videotexto? No lo sabemos, aunque creemos que posiblemente el acceso a base de datos sea su futuro más brillante. Sin embargo, no existe mucho acuerdo en este último punto, como tampoco lo existe en alcanzar un estándar a nivel mundial. Hoy en día existen tres estándares, el CEPT (Conférence Européene des Postes et Telegraphes), que supone un esfuerzo europeo, basado fundamentalmente en el sistema de mosaico, pero muy perfeccionado, utilizando muchas fuentes de caracteres fijos, caracteres modificables dinámicamente, etc., con PLPS (Presentation Level Protocol Syntax), es el sistema adoptado por los Estados Unidos y el Canadá. Este sistema utiliza las tres opciones discutidas en las líneas anteriores, haciendo quizá hincapié en la opción geométrica. Los japoneses, a su vez, muy avanzados en fotografía (y actualmente en informática) han desarrollado el sistema de codificación Captain, pensando fundamentalmente en la codificación de información procedente de imágenes fotográficas. Estos tres sistemas son absolutamente incompatibles. Sin embargo, esperamos tiempos mejores.

El problema fundamental que presenta el videotexto es la velocidad. El videotexto no dispone de movimiento eliminando así las posibles interacciones entre el usuario y el sistema. Este problema es fundamental, ya que le resta vida. Además, la mayoría de los programas educativos que intenten enseñar cualquier disciplina se basan en sistemas interactivos, ya que las técnicas modernas consideran que la participación es un elemento fundamental en el aprendizaje.

Los terminales «inteligentes» tienen resuelto el problema anterior, resultando mucho más «vivos». Por otro lado, los decodificadores de videotexto están compuestos por un microprocesador, una memoria RAM (lectura y escritura), una memoria ROM (sólo lectura) un teclado, e interfaces de comunicación, casi exactamente igual a un pequeño ordenador



Sistema informatizado para la educación de sordomudos.

doméstico. Esta identidad es muy interesante, ya que con pequeñas modificaciones, de costo reducido, se podría ofrecer un producto muy interesante, que utilizaría programas obtenidos de la base de datos del videotexto, y, como puede verse, las posibilidades serían enormes.

El sistema adoptado por los Estados Unidos y el Canadá, el PLPS, es un lenguaje descriptivo de imágenes estáticas, de forma que para producir algún tipo de movimiento no hay más solución que borrar cada pantalla y redibujar de nuevo. Otra posible solución consiste en cambiar los colores, o permutar el color del fondo con el del texto. En principio, el método puede parecer poco interesante, pero dominando las técnicas se pueden obtener resultados muy aceptables (naturalmente dependiendo mucho del artista que realice la animación, de forma que con otros medios no sería necesario que el «animador» tuviera tanta experiencia ni tuviera necesariamente que estar tan dotado artísticamente).

Sin embargo, la simplicidad del sistema es muy importante. El decodificador utilizado no tiene por qué ser demasiado complejo, y, por tanto, puede utilizarse casi cualquier modelo provisto naturalmente de memoria y generador de vídeo.

Para este tipo de tareas también existe material especializado. Por ejemplo, se puede disponer de un conjunto típico, el TI9918A, que dispone de unos registros especiales de direccionamiento que permiten almacenar «sprites» en cualquier punto de la pantalla, cambiando su posición a cualquier otra coordenada muy sencillamente. De este modo se obtiene una flexibilidad enorme, lo que redundará en productos de una viveza e interés muy grandes. Para adaptar estos sistemas a videotexto sería necesario crear los «fondos» y los «sprites» con el PLPS. Esto se hace creando unas cadenas con ciertos delimitadores, de forma que «todo» lo que se encuentre entre dos delimitadores se considerará un «sprite» que permanecerá en la posición inicial, pero será desplazable a cualquier punto de la pantalla, tal como antes indicamos.

Muchos ordenadores convencionales disponen de este conjunto TI99, por lo que resultan óptimos para juegos. La televisión ofrece anuncios en los que se ve el mismo ordenador utilizado para jugar y para trabajar. El problema fundamental es la terrible falta de acuerdo entre organismos para crear una normativa adecuada. La mayoría de los sistemas existentes son absolutamente incompatibles. Los requerimientos básicos para que el sistema de videotexto funcionara de forma cada vez más rentable y útil, serían una mayor compatibilidad, grandes bases de datos, con posibilidad de seleccionar entre muchas opciones, tecnología típica de juegos electrónicos y ordenadores «caseros» con características gráficas especiales.

Los programas de enseñanza asistida son un ejemplo típico del videotexto. Sin embargo, como antes explicamos, la enseñanza asistida por ordenador es mucho más efectiva si es interactiva, por lo que los esfuerzos se han encaminado principalmente en aumentar la velocidad y también en mejorar las presentaciones. Un programa muy interesante es el desarrollado por Robin Hight por encargo de la John Hopkins University. Es un ensayo pensado para sordomudos e incluso otras personas que tienen dificultades de audición. El sistema tenía como objetivo enseñarles a leer la fonética de las palabras en los labios. El programa distinguía diecinueve sonidos distintos, y ofrecía imágenes mostrando los dientes, la lengua y los labios. El ordenador en el que se procesaba es un sencillo Apple II de 48 K. El programa convertía las frases en dibujos con

los correspondientes sonidos. A primera vista, puede parecer imposible que se pueda utilizar una memoria tan pequeña para almacenar tanta información, más el sistema operativo, etcétera. Sin embargo, el problema fue resuelto por Hight dividiendo las imágenes en componentes, que pueden a su vez condensarse en tablas con datos. Cada vez que se necesite generar una nueva imagen, ésta puede ser creada fácilmente tomando los elementos de la tabla.

La animación del programa se obtiene por el método de memorias alternativas. Mientras se está visualizando una forma de boca en la primera página, se está dibujando la segunda en la página dos, para a continuación pasar a presentar la segunda, mientras se genera la tercera, y así sucesivamente. Con este método se puede conseguir hasta quince imágenes por segundo, que es una calidad suficiente.

El color es otro factor importante a estudiar en videotextos. El problema fundamental es que los colores suelen cambiar sustancialmente bajo condiciones distintas. Es muy distinto si se observan desde cerca, donde los contrastes son intensos, que si se ven desde lejos, donde tienden a desdibujarse. Por ejemplo, si se coloca un determinado rojo junto a un verde, el rojo se acrecentará, resultando reforzado. Si se coloca junto al amarillo, parecerá algo azulado; si es junto al azul, parecerá algo amarillento. Además, el negro hace más profundos los colores, y el blanco ilumina y aclara mucho más. Todas estas consideraciones deben tenerse en cuenta al crear páginas y dibujos. El color de fondo es también muy importante, ya que puede resultar animante o, por el contrario, resultar muy cansado si se va a trábajar varias horas con el ordenador. A

continuación, daremos unas pequeñas normas sobre colores:

El rojo y el amarillo son colores dinámicos, fuertes, que llaman la atención del espectador. Pueden, pues, utilizarse para resaltar zonas del dibujo, frases, líneas de un diagrama, etc.

El azul y el verde son colores relajantes y tranquilos. Pueden utilizarse como fondo o para frases o zonas del dibujo que no se vayan a resaltar.

El color de fondo más relajante para trabajar es el gris, compuesto por una mezcla a partes iguales de rojo, verde y amarillo. El blanco no es recomendable en absoluto, ya que cansa mucho a los ojos. El negro es difícil de obtener.

Los colores oscuros sobre fondo negro resultan muy pobres, parecen perderse.

Colores de una intensidad y brillo semejantes situados próximos ofrecerán un poco de contraste, pero, por el contrario, serán desencansantes para la vista.

Los colores de intensidad y brillo muy diferentes ofrecerán contrastes acusados, llamando la atención de la vista, pero resultando cansados.

Para realizar presentaciones llamativas, es útil seleccionar un color de texto fuerte (moderadamente) sobre el color de fondo complementario.

Las figuras tienen mucha más entidad si están bordeadas por algún color que contraste con la propia figura.

Los «artistas» de la animación dominan estas y muchas más técnicas, relacionadas muchas veces con la fotografía, y estamos seguros que, con el tiempo, la animación y presentación de imágenes en el monitor llegará a ser un arte más.

Captain. Sistema de codificación de gráficos empleado por los japoneses que utiliza técnicas fotográficas. Es especialmente eficaz, por esta razón, en el manejo de gráficos obtenidos a partir de fotografías.

Caracteres DRCS. (Dinamically Redefinable Character Set). Juego de caracteres redefinibles dinámicamente. Los sistemas que utilizan estos caracteres, junto con los caracteres fuente básicos del sistema, aportan una mayor flexibilidad en el momento de diseñar los gráficos.

Caracteres fuente. Conjunto básico de caracteres utilizados en la preparación de un gráfico o de un texto. Suele estar almacenado en memoria de un modo permanente (aunque en ocasiones es modificable). En contraposición a este juego básico, se suele utilizar otro conjunto de caracteres variable definido en cada momento por el diseñador.

Caracteres geométricos. Juego de caracteres gráficos que se construyen a partir de figuras geométricas elementales: rectas, segmentos, arcos, sectores y segmentos circulares polígonos, etc.

Caracteres mosaico. Juego de caracteres gráficos especiales, que complementan el juego estándar, contruidos mediante la división del espacio de un carácter en pequeñas zonas, coloreables de forma independiente.

CEPT. Conférence Européenne des Postes et Télégraphes. (Conferencia Europea de Telégrafos y Correos). En videotexto se utilizan estas siglas para designar el estándar definido por dicha conferencia: básicamente un sistema de mosaico muy perfeccionado.

Color de fondo. Color de base que adopta la pantalla en presencia y sobre el que se dibujan los gráficos y texto.

Color de texto. Suele designarse con este nombre, al color con el que se escribe el texto sobre el color de fondo, y también, por extensión, al color utilizado en el trazado de dibujos y gráficos.

Decodificador. En videotexto, interface que obtiene la imagen presentable en pantalla, a partir de la información codificada recibida. Es uno de los componentes de la estación receptora de videotexto.

Fuente. (Ver **Juego de caracteres fuente**.)

PLPS. Presentación Level Protocol Syntax. Sintaxis del Protocolo para la Presentación de Gráficos. Estándar muy difundido en los Estados Unidos y Canadá, y que se basa en técnicas mixtas de otros sistemas varios.

Progresiva, transmisión. Sistema de transmisión específico, con una codificación limitada, utilizando el envío de gráficos.

Sistema de animación. Procedimiento para dotar de movimiento a las figuras definidas estáticas, en cualquier conjunto de gráficos.

Sistema fotográfico. Sistema basado en un barrido de la pantalla punto por punto para recabar la información que aparece en cada punto constitutivo de una imagen gráfica dada.

Sprite. Figura elemental predefinida, con la que se construyen gráficos complejos. Estos sprites se almacenan de modo permanente durante el diseño, y son colocados en diferentes puntos del gráfico según se vayan necesitando. A veces se utilizan para designar a los sprites los términos castellanos «fantasma», «hombrecito», «monigote», etc.

VOCABULARIO DE INFORMATICA

Lenguaje máquina. Lenguaje del más bajo nivel, de entre los utilizados por el ordenador. Es interpretado directamente por su CPU, y utiliza muy eficazmente todos los recursos de hardware.

Lenguajes de programación. Conjunto de caracteres y reglas con los que se puede escribir un programa de ordenador. Entre los más conocidos están el BASIC, PASCAL, FORTRAN, COBOL, FORTH, ADA, etc. Suelen denominarse también lenguajes artificiales.

Less equal. Término inglés para designar menor o igual que. Suele indicarse con los símbolos $< =$.

Less than. Término inglés para designar menor que. Suele indicarse con el símbolo $<$.

Letra. Símbolo gráfico que utilizado solo o combinado con otros símbolos representa palabras de un lenguaje escrito. No deben confundirse las letras con otros símbolos del lenguaje como comas, etc., llamados caracteres no alfabéticos.

Level. Término inglés para designar nivel o grado de subordinación dentro de una jerarquía.

LF. Siglas que representan la palabra inglesa Line Feed, que significa salto de línea.

Library. Término inglés que significa biblioteca. Véase, pues, biblioteca. En ocasiones suele utilizarse el anglicismo librería.

Librería, rutina de. Rutina probada con anterioridad y almacenada en una librería o biblioteca de programas.

LIFO. Siglas correspondientes a Last In, First Out, último en entrar, primero en salir. Método de control de una cola de espera para proceso. El sistema seguido es que el último que llega es el primero que se procesa. (Sucede como en un tubo de almacenamiento, cerrado permanentemente por un extremo, lo primero que se saca es lo último que se introdujo.)

Línea. Término que puede referirse a muchas cosas. Por ejemplo, línea de retardo, línea acústica, línea de referencia, línea de flujo, etc.

Lineal, optimización. Método básico de optimización utilizado en programación lineal.

Líneas, impresora de. Dispositivo que imprime a un mismo tiempo todos los caracteres de una línea. En contraposición con impresora de caracteres.

Linker. Programa que realiza la unión entre varios programas, rutinas o subprogramas. A veces se nombra por el término castellano equivalente: encadenar. También se utiliza como sinónimo el concepto de «montador».

Linking. Término inglés que indica un procedimiento por el cual se unen distintas partes de un programa o varios programas uti-

lizando para ello otro programa, al que se denomina linker.

List processing. Término inglés para designar un proceso de los datos en forma de lista. El orden lógico de los elementos se puede cambiar, por lo general, sin alterar en absoluto sus posiciones físicas respectivas. Existen lenguajes específicos para el manejo de listas (LISP, LOGO...), aunque se pueden utilizar en estas tareas también los lenguajes de uso general.

Lista. Colección ordenada de elementos. También puede referirse a un listado de ordenador.

Lista de espera. Conjunto de programas o rutinas que están esperando para ser procesados.

Load. Término inglés que significa cargar, introducir elementos en los sistemas de almacenamiento o en los registros de trabajo.

Load and go (cargar y ejecutar). Técnica operativa en la que no existen detenciones entre las fases de carga y ejecución de un programa, y que puede incluir también ensamblaje y compilación.

Load and store (cargar y almacenar). Términos ingleses que designan un modo de almacenamiento no muy utilizado hoy en día. Consiste esencialmente en que la carga del programa y su posible compilación y ensamblaje se realizan y concluyen por almacenamiento del código objeto.

Load, modo. En algunos ordenadores que pueden aceptar longitudes distintas para las variables, transmisión que mueve los datos junto a ciertos delimitadores. En contraposición al «modo move».

Loader. Término inglés para designar un cargador, es decir, un programa que sirve para cargar en la memoria central otros programas. El cargador que carga el primer programa, nada más conectar el ordenador, se designa con otro término inglés, bootstrap, aunque a veces también se llama IPL (Initial Program Loader).

Local. Término muy general, que puede aplicarse a muchos términos. En informática se

utiliza en contraposición a global. Puede aplicarse, por ejemplo, a variables (es lo más común), a redes, a transmisiones, etc.

Local area network (LAN). Término inglés que designa una red de área local, es decir, una red de comunicaciones que enlaza varias estaciones definidas en modo local, es decir, establecida dentro de un mismo entorno, edificio, etc.

Local, modo de eco. Devolución de la información a su fuente primitiva que se produce de forma local.

Local, optimización. Producción de código objeto óptima en algún sentido. Generalmente, se dirige a explotar de la mejor forma posible las características especiales de la arquitectura de la máquina.

Local, red. Red de comunicaciones formada por varias estaciones establecidas en un ámbito local. En contraposición a la red de comunicaciones remotas establecidas a través de la línea telefónica.

Location. (Ver **Posición**).

Lock. Término inglés que indica bloqueo o cerrojo. Se utiliza fundamentalmente como medio de protección, por ejemplo, en un archivo al que se puede acceder, pero no modificar, etc.

Lockout. Término inglés para indicar el bloqueo. Suele referirse a un mecanismo que se establece para controlar el acceso a un recurso compartido.

Locks and keys. Término inglés que designa un sistema de memoria en el que cada uno de los segmentos establecidos en ella tiene asignado un número, al que se puede acceder desde el sistema operativo, obteniendo la autorización correspondiente. Es, en suma, un sistema de protección de la información.

Lógica. Ciencia del razonamiento. Estudia la verdad o falsedad de las proposiciones que son la expresión del pensamiento. La lógica matemática utiliza medios matemáticos de discernimiento. En informática, cuando se habla de lógica, suele entenderse lógica booleana.

Lógica, instrucción. Instrucción que ejecuta una operación lógica en álgebra booleana. Utiliza los símbolos AND, OR, NOT, etc.

Logical. (Véase **Lógica**.) También se utiliza este término, puesto de moda por los franceses, para designar al conjunto de programas que gobiernan el funcionamiento del ordenador: lo que los americanos llaman software.

Lógico. Término que indica un concepto, algo virtual, en contraposición a físico, o real. Referente a la lógica.

Lógico, diagrama. Diagrama que representa un diseño lógico, o la estructura general de un proceso. En algunos casos a la implementación de hardware.

Lógico, diseño. Especificación de las relaciones existentes entre las partes de un sistema, en términos de lógica simbólica, y sin considerar de forma inmediata la implementación de hardware.

Lógico, elemento. Dispositivo que realiza una función lógica, en contraposición a los dispositivos físicos del sistema.

Lógico, símbolo. Símbolo utilizado para representar un elemento lógico de forma gráfica.

Lógico, operador. Operador que chucu chucu chucu.

Lógicos, tipos de datos. Tipo de datos abstractos. También puede referirse a datos que incluyen valores verdaderos o falsos

(es decir, datos a manejar en lógica booleana).

Lógico, valor. Valor booleano, es decir, cualquiera de los valores verdadero o falso que se aplican a una proposición. Se pueden indicar utilizando un único bit, aunque es frecuente utilizar un byte.

Logo. Lenguaje de programación de alto nivel, pensado fundamentalmente con fines didácticos. Fue desarrollado por Seymour Papert. Es un lenguaje funcional con gran facilidad para el manejo de listas y predisposición para el diseño gráfico en pantalla.

Longitud. Término que se utiliza para indicar el número de elementos de un conjunto. Suele aplicarse a palabras, caracteres de una cadena, número de registros de un archivo, etc.

Loop. (Véase **Bucle**.)

LPM. Abreviatura de líneas por minuto. Indica la velocidad de impresión (para las impresoras que escriben por líneas, frente a caracteres).

LSD. Siglas de Least Significant Bit. (Véase **Bit menos significativo**.)

LSI. Siglas de Large Scale Integration-Integración a gran escala. Se utiliza para indicar una moderna tecnología de fabricación de elementos electrónicos en la que los componentes son pequeñísimos, y se pueden introducir circuitos de hasta 10.000 transistores en un único chip o pastilla.

